Psychological Issues of
Human Computer Interaction in the Work Place
M. Frese, E. Ulich, W. Dzida (Editors)
© Elsevier Science Publishers B.V. (North-Holland), 1987

3

A THEORY OF CONTROL AND COMPLEXITY: IMPLICATIONS FOR SOFTWARE DESIGN AND INTEGRATION OF COMPUTER SYSTEMS INTO THE WORK PLACE [1]

Michael Frese

Dept. of Psychology, Ludwig-Maximilians-Universität München,
Leopoldstr. 13, D- 8000 München 40, Federal Republic of Germany

This article argues that control over one's activities and the working (system) conditions, complexity and complicatedness can be differentiated. Control refers to decision possibilities and efficient action, complexity to decision necessities, and complicatedness to those decision necessities that are difficult to control and socially and technological unnecessary. Control should be en-hanced in software systems, complexity should be opti-mized, and complicatedness reduced. When control stands in contrast to non-complexity (ease of use) and to some "intelligent" or adaptive features of software, control should be increased at the expense of other features because control has long term positive consequences on stress-effects and performance.

INTRODUCTION

In this article, I want to discuss three concepts, control, complexity and complicatedness. I want to argue that complicatedness should be minimized, complexity optimized, and control maximized. Control should be maximal because having control over conditions and one's actions leads to positive motivational and cognitive consequences. Lack of complexity leads to boredom and monotony, too much complexity to overload. Complicatedness leads to the feeling of nuisance and loads mental capacity with processing needs that are not task oriented.

The approach taken here is in opposition to some of the literature on human computer interaction that regard as main solutions for increasing "user-friendliness" and "usability" of computer programs the reduction of their complexity (e.g. Card, Moran & Newell, 1983; Polson & Kieras, 1985; Shackel, 1985). It is interesting to see, of course, that, indeed programs that do seem to reduce complexity (e.g. the Macintosh programs) are often applauded by the users for their ease. People in their everyday life reduce the complexity of their situations as well. In work life they do this, for example, by not paying attention to things that are not quite so important (e.g., even experts only know about 25- 35% of the commands of complex systems), by chunking smaller parts into larger ones (e.g., in programming where parts are worked out as subroutines that one can worry about later), through holding open options rather than planning the whole way from the start (Oesterreich, 1981), by psychological automatization, so that one has central processing capacity free to think of other things (Hacker, 1978). Thus, people seem to be complexity-reducing beings.

On the other hand, there is a very old tradition in industrial psychology -- the job enrichment, humanization of work or industrial democracy literature -- that suggests that one should increase the complexity of the jobs (Emery & Thorsrud, 1976, Hacker, 1985, Hackman, 1977, Ulich, Groskurth & Bruggemann, 1973) (actually, they argued for higher control and higher complexity but usually did not differentiate between these two concepts). It was the promise of these job restructuring attempts that performance and well-being would be increased by giving people jobs that were appropriately complex and needed the qualifications of the workers.

Thus, these two traditions, the software ergonomics and the humanization of work protagonists, recommend very different approaches. Software ergonomists argue for a reduction of complexity; humanization of work advocates argue for an increase in complexity. Both seem to have data supporting their positions. Who is right?

It is the argument of this article that the crucial category is control and that aspects of the system that ought to be weeded out are aspects of complicatedness, but not of complexity in general. Complicatedness is that part of complexity that is difficult to control. Thus, easiness is not an important criterion for evaluating computer systems in and of itself but it is dependent on control whether easiness of computer systems leads to better or worse performance and well-being.

I shall first develop the concept of control. Then I shall develop the notions of complexity and complicatedness. The next step is to compare and contrast control, complexity and complicatedness. Then I shall argue that control at work, complexity, and complicatedness have repercussions on stress-effects and performance. Finally, I can use these concepts and apply them to software design. When developing the concepts, I shall talk about control and complexity in general, as it relates to the work situation because the design problems of computer systems are not qualitatively different from the design problems of work places in general.

THE CONCEPT OF CONTROL

Experiencing control means to have an impact on the conditions and on one's activities in correspondence with some higher order goal (Frese, 1978). This impact may be potential or it may directly influence the conditions. Potential control was studied by Glass & Singer (1972). In their experiments the subjects had a button that could turn off a loud noise (the stressor). The subjects were, however, asked not to use this button (and all of them complied). This condition produced less stress than not having such a control button. Direct control in stress situations was the issue in Seligman's (1975) experiments. When there was no control, helplessness developed.

Control at work may be applied on an individual or on a collective level. For example, if a team of two persons are able to use a certain software system but each individual alone could not have done it, they have collective control.

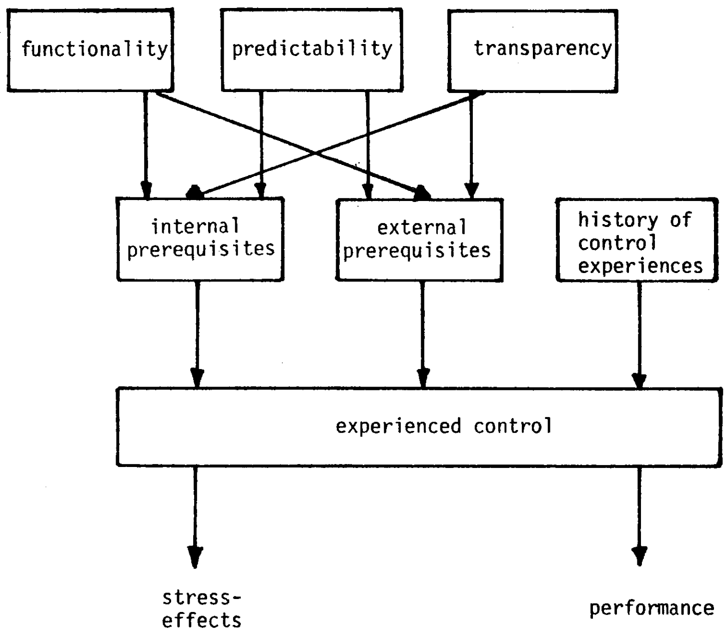

Figure 1: The conceptual framework

It is important to emphasize the goal-related nature of control. Without a goal, there is no issue of control. This concept is somewhat different from other definitions, e.g. the one by Seligman (1975) in which control is defined as non-contingency of events (an event appears or disappears regardless of a person's actions). Such a definition, not bound to the goals of a person, produces some conceptual difficulties. According to this definition, a person would have control, if he or she accidentally (contrary to the intention) produces a serious mistake (like breaking a vase). According to my concept of control, the person did not have control in this case.

Figure 1 functions to describe the theory of control that underlies this article. Experienced control has a direct influence on performance and stress effects. Experienced control is influenced by internal and external prerequisites. External prerequisites imply decision possibilities of the system and at work; internal prerequisites refer to an (appropriate) mental model and adequate skills vis-a-vis the system. These internal and external prerequisites are themselves again influenced by the functionality, the transparency, and the predictability of the system. Experienced control is also influenced by earlier experiences of control. Earlier experiences of control are a significant issue in occupational socialization (Frese, 1982) but are not important in our context here and will, therefore, not be discussed any further.

External Prerequisites

Table 1: Aspects of External Prerequisites of Control

Action sequence	Decision possibilities		
	Sequence	Timeframe	Content
Tasks (Goals)			
Plans			
Feedback (Signals)			
Conditions			

A system is controllable when a person has influence over his actions and over the conditions under which he acts. An action consists of a certain sequence (this sequence is variable, of course): goal development and goal decision, plan development and decision, execution of the action and use of feedback (Frese & Sabini, 1985, Norman, 1986). Having influence means to be able to decide what goals, what plans, what kind of feedback a person is using under what conditions. If the environment does not provide the freedom to decide, the person does not have any control. This concept is summarized in Table 1.

Decision possibilities can appear with regard to sequencing, timeframe, and content of tasks, plans, use of feedback and conditions. Decision points with regard to sequence may for example mean that people are able to determine which task they do first and which one second, in which sequence plans are being formed and executed, or in which sequence they call on signals to inform them of the success of their activities. Timeframe refers to two sets of decision possibilities: First, the decision, when a certain task is tackled or a certain plan is performed; second, deciding on how long it will take to work on a task or on a plan. Similarly, the timing of the signal alludes to when it will appear and how long it is displayed. Content refers to the substance of the decisions with regard to task, plan, signal and conditions: What particular task is done, what plan is formed, what kind of signals does one choose to use and what conditions exist for work.

Four issues are worth emphasizing in this context. First, the decisions must refer to a goal (or in the case of a task, to a superordinate goal). I decide with some goal in mind. As long as something is impertinent to the goals, non-control does not matter. For example, not being able to control the weather has few negative consequences because we usually do not develop goals in this regard. However, not being able to control the temperature in the room might have negative consequences because people have goals in this area. Similarly, if I am just trying out a new computer system without wanting to do something specific with it, control does not matter.

One implication of this reasoning is that control has dynamic properties. Computers enhance potential controllability towards a variety of objects (e.g. being able to correct errors before they appear on paper, being able to check calculations more easily, etc.). This leads to a proliferation of new goals. Using a typewriter, I cannot decide to exchange paragraphs without cutting and pasting -- non-control does not matter. But once I use a word processor, not being able to exchange paragraphs, matters. Thus, the more people know about computers and their potential and the more can be done with computers, the more goals are developed and the more important the chances to decide will become.

Second, freedom to decide has a positive quality only when the decisions do not involve high risks. If all the alternatives involve high risks, then controllability may lead to an aversive situation. This is related to the issue of goal again, because we usually do not develop goals that are very risky. Again, this has important consequences in human-computer interaction. For example, the risk of losing the whole text with one command, may be such a high risk situation that makes the situation aversive for the novice even if there is potential control over the decision points.

Third, there is a hierarchy of goals within any one person at any one time (there may be, of course, multiple and also conflicting goals). The higher up in this hierarchy a particular goal is, the more important and central the decision becomes (e.g. life or career plan decisions). Thus, control and non-control must be weighted by the importance of the goal. However, a caveat has to be made here: For most people, higher order goals are not at their disposal any more after they have been committed to them for some time. For example, a life goal cannot be redecided at any one time. Therefore, the more practical day-to-day activities stand in the foreground of one's attention. There is actually little need to consciously think about the higher level goals. After a while, this may lead to a reduction of their conscious "importance" and lower-level goals are in the foreground of attention and thinking and achieve high priority.

Fourth, aside from the above points about the importance of goals and goal hierarchy, it is important to ask the question, how much of the time a person is exposed to non-control or control situations. The theory predicts that exposition time is an important variable (Frese, 1984). If one is constantly under conditions of non-control (little decision making power) even in small matters, there is an impact even if the goals are not very important. The longer, one is exposed to these conditions, the stronger is their impact on experienced control, even if higher order goals (like life goals) are still to be influenced by the individual.

Internal Prerequisites of Control

The external condition represent only *potential* decision points with regard to some goal. To realize this potential, one needs to have knowledge and skills as internal requirements. The former refers to metaphors (Carroll & Thomas, 1982), conceptualizations or mental models of the system (Rouse & Morris, 1986, Norman, 1983). Skills are particularly important as prerequisites of control. Skills are related to efficient actions (Volpert, 1974, Semmer & Frese, 1985), which means that goals, plans and the use of feedback must be realistic, stable-flexible and organized (compare Table 2). The goals and plans must be realistic in terms of timeframe and sequence. Stability implies that goals and plans are not immediately given up when there is some negative feedback. On the other hand, plans should for example be executed in a flexible way, i.e. they should be adjusted to the environmental conditions. Finally, organization implies that those plans that are used very often under redundant circumstances should be psychologically automatized so that one has central processing capacity free to deal with other work demands.

Table 2: Skills

Action sequence	Realistic	Stable-flexible	Organized
Goals			
Plans			
Use of feedback			

Functionality, Transparency, and Predictability

Functionality, predictability and transparence impinge on the external and internal prerequisites of control.

Functionality refers to whether a computer program allows and enhances the completion of a task. One issue of functionality is, for example, whether the computer system models real world tasks. Thus, a statistics program should calculate correctly and do what it is supposed to do. A spell program should have enough words in the dictionary. Without functionality there is no control, because the decisions are not meaningfully related to the goal (task) any more. However, a high functionality does not necessarily imply, that there are decision points available.

Transparency implies that the user can develop an internal model of the functions of the system (Maass, 1983)and that he can develop the necessary efficient skills. Thus, the system should not confuse the user by giving different commands to do the same thing under different modes or by giving explanations that are inconsistent or only half true. At each point it must be clear to what end the system is doing something and what the reasons are that the system behaves the way it does. The problem of transparency comes up particularly with so-called intelligent systems that change with the user and may thus confuse the user with ever- different procedures. Under conditions of non-transparency, the user cannot make adequate decisions referring to his goal and he cannot develop a realistic conceptualization and a flexible and stable approach. Transparency is not identical to control because it is possible to develop a system that is completely transparent, nevertheless offer little control (e.g. an expert system that perfectly explains why it is doing what, but that does not allow users' decisions).

Predictability has some overlap with the concept of transparency. If a system is not predictable, it is most likely not transparent. However, transparency refers to the present, predictability to the future. If a system's behavior cannot be foreseen it is not predictable. In the context of predictability, two issues have been discussed: predictability of what (pwhat) and predictability of when (pwhen) (Miller, 1981). Pwhat implies that one knows what happens when a certain command is typed into the machine. Pwhen implies that one knows when a certain command is executed and completed. The latter is, of course, related to the issue of system response time (Boucsein, Greif & Wittekamp, 1984). When the response time is variable, there is little predictability.

The relationship between predictability and the internal and external prerequisites of control is complex. It is possible that a system is predictable but not controllable (e.g. everything is determined by the system but there is a signal that tells what will happen next). It is also possible (although unlikely) to conceive of a system that is controllable but not predictable. For practical purposes, lack of predictability makes decisions meaningless, because when the states of a system cannot be foreseen, one cannot make adequate decisions. Similarly, it is related to the internal prerequisites of control because it is hard to develop a concept of an unpredictable system and one cannot develop an organized approach towards it.

COMPLEXITY

Our approach to the concept of complexity is shown in Table 3. Again, decisions can be related to goals, plans and signals (signifying some action, as in the case of menus) or feedback (after some action, e.g. in the case of error messages). In contrast to control, the decisions refer here not to decision possibilities but to decision necessities. There are three sets of conditions leading to complexity (Table 3) (cf. Dörner, 1976):

(1) A high number of different goals (or subgoals), plans (or subplans), and signals that need to be sequenced, put in some timeframe, or which have to be connected in terms of content.

(2) A high number of relationships (a) within each cell, e.g. relationships between different goals and (b) between cells, e.g. relationships between goals and plans. An example for (a) is that one command has implications for using another one; an example for (b) is that one has to know for which goals one can use a specific command.

Table 3: The Concept of Complexity

	Decision necessities		
	sequence	timeframe	content
Number of	goals / plans / signals	goals / plans / signals	goals / plans / signals
Number of relationships between	goals / plans / signals	goals / plans / signals	goals / plans / signals
Number of conditional relationship between	goals / plans / signals	goals / plans / signals	goals / plans / signals

(3) Not only the sheer number of the relationships determine complexity but complexity is increased if there are a large number of conditional relationships, e.g. that one type of command has different consequences (e.g. different error messages) in different modes or that two different plans are influenced by a third one (a kind of moderator or suppressor effect).

The various aspects of complexity are interrelated, of course. The number of goals, plans, and signals determine the potential number of relationships. The more relationships there are, the more likely it is that there are conditional relationships.

Complexity defined in this way is neither a characteristic of the environment nor of the person alone. It is a characteristic of the interaction of the person with the environment. For example, the novice still has to make decisions that are remembered facts for the expert.

In summary, complexity is determined by the sheer number of decisions that have to be made and by the relationships of these decisions. Thus, there is decision necessity.

Experienced control is similarly related to decisions, but it implies a reference to the individual's goals. Thus, there is decision freedom. One implication of this is that a person may be forced to do a complex task that he or she does not want to do -- then control is low but complexity is high.

My concept is different from Kieras & Polson's (1985) concept of complexity in several ways. Although Kieras & Polson have a similar general definition of complexity in their specific experiments, they seem to be concerned only about the number of elements in a system and not about the relationships between these elements (e.g. Polson, Muncher & Engelbeck, 1986). However, the question of relationships between the elements is important, otherwise one could not explain why delayed feedback cycles are more complex than non-delayed feedback (Dörner, 1986). Furthermore, their approach has difficulties explaining the differences between a novice and an expert. In our view, the number of elements that are acted upon are the same, but the number of decisions are different: An expert does not have to decide any more, how to delete a character: He already knows it and it is probably already automatized, i.e. no more thought (and decision) has to be wasted for doing this particular operation. (It would be possible, of course, to integrate this into their concept, as the example of Anderson's, 1983, production system shows.) The most serious problem in Kieras & Polson's formulation, however, is that it does not distinguish between complexity and complicatedness; moreover, I doubt that a low-level approach like theirs is able to make such a distinction.

COMPLICATEDNESS

Complicatedness partly overlaps with complexity. My argument here is that complicatedness is to a large extent complexity that is difficult to control. Thus, control decides whether something is just complex or additionally complicated. A system becomes complicated when it is complex and when one of the following additional conditions apply:
- when there is little functionality
- when there is high intransparency
- when there is high unpredictability
- when there are fewer decision possibilities than necessary
- when there are more decision necessities than are to be comprehended by the mental model or executed by the skills

- when the complexity is neither socially nor technologically necessary or adequate.

It is obvious that the first 5 reasons are related to the concept of control, as it was developed in Figure 1 (the last one has to be treated separately).

When there is lack of *functionality*, the worker has to work around the problem. Workers often experience that they have to "cheat" the system, to get the required result. In computer aided storage, they might have to cheat on how many parts are still in storage to get the computer to order them in time (Dr. Rödiger reported this example to me). Gasser (1986) has reported on the "ubiquity of anomaly" when using computer systems in office environments and how strategies of "fitting", "augmenting", and "working around" have to be used to make sure that functionality is achieved. In any case, these are complicated procedures.

When there is *intransparency*, the worker cannot develop an adequate model of the system. Intransparency can come about, when the system changes while a person is working with it. This is a problem with so called adaptive systems that change e.g. with the user's experiences (e.g., Chin, 1986). The changes may make the system intransparent, even for the designer himself (Fitter & Sime, 1980). Similar problems appear, when "intelligent" systems are able to learn new procedures -- any state is then intransparent, not controllable and, therefore, complicated to use.

When there is *unpredictability* (pwhen and pwhat), planning for the timeframe is difficult when the timing is not predictable and planning for what one is doing and forecasting what will happen is complicated. I already alluded to the problems associated with a variable (and long) system response time. Again, problems of unpredictability of what will happen can arise in adaptive and "intelligent" systems because of their changing status. The problems of imcompatibility can be seen within the framework of unpredictability as well: Incompatibility means that the expectations formed in one part of the system (or with other systems before) do not lead to the right predictions. Incompatibility may, for example, come about when a metaphor is not consistently used throughout (Rohr & Tauber, 1985) or when certain commands in one mode mean something else in a different mode.

The lack of *decision possibilities* is an external prerequisite of control (Table 1). When there are not enough control possibilities, the system is seen as complicated, because one has to work around things (see above) and because it is more exhausting (stressful) (cf. also the chapters by Ackermann & Ulich, Corbett and Spinas in this book). Hacker (1983) shows that if the sequence of a task is given, the worker shows more stress and works less planful than when the sequence can be chosen by the worker. Benbasat & Wand (1984) argue that a system-guided dialogue may be preferred by the novice but is a nuisance for the experienced user. This fits nicely into our theoretical framework. For the novice, complicatedness is related to the lack of adequate mental models and skills. Therefore, a system-guided dialogue helps the novice develop a mental model and the skills necessary. However, once the internal prerequisites are developed, further system-guidance becomes a nuisance and the added number of steps that one has to go through are interpreted as complicated. Similar problems appear with menu systems in which the experienced worker is not allowed to work with commands alone (and is, therefore, forced to plow through an endless array of menus and prompts and questions that he knows very well). In this case, the possibility to chunk information is not allowed by the system.

Whether something is seen as complicated depends on how much one has already learned of the system. These aspects are, of course, related to *mental models and skills* (cf. Table 2). If there are more decision necessities than comprehensible and if there are more elements than can be fit into the working memory, a system is seen as complicated. There is an additional distinction: (1) A system may have a high number of elements that are purely filling up memory. This is true, when e.g. the starting procedure for a system implies the execution of 10 different operations. In this case, technological automatization would increase control and reduce complicatedness. (2) Another set of a high number of elements can be reduced by thougthful patterns. Here thinking and even creativity may be required to reduce the number of elements. While both types may be perceived by the novice to constitute complicatedness, only the former (1) will also be seen as complicated by *both* the novice and the experienced. Hacker (1983) found that the number of elements that had to be kept in working memory made for a high number of errors and fatigue while the complexity that leads to thinking and creativity decreased fatigue.

The reasons discussed so far are related to the issue of control, the following goes beyond it. Complexity is seen as complicated when it is not *socially or technologically adequate.* The notion of technological adequacy implies a dynamic property. While some people were enthused some time ago about the line editor because it gave editing possibilities (thus more control), it would now be perceived to be complicated. An implication of this is, of course, that people who know the newest technological developments will be most prone to judging older products as complicated. Similarly, something is also seen as complicated when complexity is perceived to be the product of malicious intent -- e.g. when a co-worker put into the data some complications by design so that other people cannot work with them.

Thus, a system is seen as complicated when its complexity cannot be controlled because of extrinsic aspects of the system or intrinsic reasons of having learned not enough about it. Complexity itself helps to make a system interesting and not monotonous but if this complexity leads to decision necessities that cannot be affected, then it becomes a complicated nuisance. Note that this concept of complicatedness has a social dimension -- something that is seen as "not necessary" "out of line", etc. is complicated. Thus, the same system may become complicated with social and technological advance.

This reasoning may allow to solve the above mentioned differences in recommendations given by the humanization of work school and by ergonomists. The humanization of work advocates were mainly concerned with complexity (and partly with control) enhancement but were not concerned with complicatedness. The ergonomists are often working on issues of reduction of complicatedness (rather than complexity per se). I contend that the reasoning presented here, helps to untangle these issues and develop a more sophisticated approach to software design. Before we go into this, it is necessary to discuss the effects of control, complexity, and complicatedness, however.

THE EFFECTS OF CONTROL, COMPLEXITY, AND COMPLICATEDNESS

Control has been shown to be related to stress-effects directly and as a moderator. In one group of studies, people who had little control at work showed more signs of psychological and psychosomatic dysfunctioning, e.g. depression, psychosomatic complaints, irritation/strain, exhaustion, anxiety, consumption of pills, sick days, and low self esteem (Caplan et al., 1975, Dunckel, 1985, Frese, Saupe & Semmer, 1981, Gardell, 1971, Karasek, 1979, Kohn & Schooler, 1982, Kornhauser, 1965). This also holds for studies of office workers with computerized office equipment (Cakir, 1981, Smith et al., 1981, Schardt & Knepel, 1981, Turner & Karasek, 1984). A second group of studies showed that control had a moderator effect: stress had a higher impact on psychosomatic complaints (Frese, 1984, Semmer, 1982) and on death from heart attack (Karasek et al., 1981) when control was low and a low impact when control was high.

Control also affects performance. If one is repeatedly in situations of non-control, passivity increases and an active, planful and goal-oriented approach is reduced. Non-control implies that one does not have to develop one's own goals and plans of action. There are two explanations why this afffects performance: A cognitive and a motivational account. The notion of action style (Frese, Stewart & Hannover, 1987) might give a cognitive account, why non-control leads to passivity and reduced performance. Two action styles have been studied in particular: planfulness and goal-orientation (Frese et al., 1987). These action styles function similarly to meta-cognitions (Brown, in press, Gleitman, 1985). When work does not allow long-term decisions, one gets used to not plan ahead. This may become generalized and then general planlessness ensues.

Decrements in performance appear in those tasks that require planning. A similar reasoning applies to goal-orientation and also to the use of feedback.

The motivational account is related to the concept of helplessness (Seligman, 1975). If one is repeatedly in situations of non-control, one does not care to develop goals and plans because one knows that they will not have an effect on the environment anyhow.

Not all aspects of the job are equally affected by non-control. It is useful to distinguish between officially prescribed primary tasks and other secondary tasks. The latter are affected more. The primary task of the typist is, for example, to copy the manuscript without errors. A secondary task might be to correct spelling mistakes by the author, to correct errors in grammar, and to tell the author what part of the manuscript was difficult to understand. When working with computers, it is always necessary to adapt the system to the specifics of the job. Correcting and detecting software errors (or at least to work around them, once they appear), knowing what to do when there is a breakdown and organizing one's work in an efficient way are aspects of secondary tasks. In the case of full screen editing, the typist may set the tab to move around the screen more quickly or to write tables more efficiently than to just use the cursor control keys. Often performance differences between people are due less to performance differences in the primary task than in the secondary task (Hacker, 1978, 1985). In the case of typists, correcting errors takes much longer than typing the words in the first place. Thus, preventing errors increases productivity more than simple typing speed.

Thus, there are various negative effects of non-control on performance and well-being.

Lack of *complexity* leads to boredom and has similar stress-effects as control.[2] A certain amount of complexity has to exist, to be able to use creative solutions that require intellectual capacities. If jobs provide only little complexity, this leads to a reduction of the use of intellectual resources and eventually to a sort of cognitive atrophy in which one looses one's intellectual abilities to solve problems (Kohn & Schooler, 1982). On the other hand, complexity that is too high is too difficult to deal with and produces qualitative overload (Kahn, 1974). Furthermore, performance is low because there are too many decisions involved that overload central processing capacity. Thus, in contrast to control, there is an optimal degree of complexity (I am talking about complexity that is potentially controllable). Complexity that is too high stifles performance, too low complexity does the same thing. It is known from the achievement motivation literature (Heckhausen, 1980) that people like to solve moderately complex problems and that achievement is lower with very low and very high aspiration levels. Thus, emotional and performance effects are most positive under conditions of optimal complexity.

One could argue that complexity is non-functional when it is a state of the tool and not a state of the main task (cf. Hacker in this volume). When writing an article, the word processor would be the tool (and should be as simple as possible) to be able to work on the content of my article (the main task, that should be moderately complex). There is some truth to this argument and, in fact, the complicatedness of software leads to being a nuisance when it appears as part of a tool (cf. Dzida in this volume). This nuisance is related to control, again, since the complicated tool is a kind of forced detour on the route to the main goal (solving the task). However, this view does not give the whole picture for the following reasons:

(1) The differentiation between tool and main task is difficult to uphold. What is the tool and what is the main task for a data entry typist? Here the use of the tool is the main task. Similarly, my thinking while writing an article is a tool as well. At the same time thinking the content of the article through is the main task. Again, the differentiation between tool and main task is blurred.

(2) The complexity of a system should be seen within the context of the whole job. This leads to the interesting hypothesis that people whose main task is not very complex, might be stimulated by the complexity of the system. For example, data entry workers, whose job content provides very little complexity, might profit from a complex system more than managers (note: it is more useful, of course, to mix data entry work with other activities to not have this kind of low job content).

(3) With the advent of computers in the work place, many tasks are becoming computer related. Here, the main tasks can only be solved well, when one is using the computer well. This point is driven home when newly introduced computers have to be augmented and fitted or worked around (Gasser, 1986). The tools (computer) and the main tasks are mixed more and more.

(4) Computers are different from other tools, because they are multipurpose machines (DiSessa, 1986). Since they allow flexibility and adaptability, people develop the aspiration level of realizing these potentials. Thus, taking away artificially the whole complexity of the machine, will only lead to a less functional use of it.

(5) Complex tools may be thoroughly enjoyable as in the example of some video games (again the main task is to work on the tool). Complicatedness does have a meaning here, namely it refers to that part of the video game that cannot be influenced by increasing skills (of course, video games try to be complex but not complicated).

Thus, the differentiation between tool and main task may have some value in reconciling the ergonomists' and the humanization of work advocates' views mentioned in the introduction. But we need to search for an additional answer. This answer is, of course, that one has to differentiate between controllable complexity and one that is difficult to control

-- the latter I call complicatedness. While it is okay to increase controllable complexity, complicatedness should be weeded out.

I argue that *complicatedness* has none of the positive effects of optimal complexity and maximal control. When there is complexity that we have little control over or that we think is technologically and socially not necessary, we are typically angered by complicated systems and find them a nuisance (cf. also Osterloh, 1983). The important reason for this is that little control and complexity are combined. We are forced to make decisions, take care of processes, go detours that do not lead directly to our goals, that are not stimulating and that are not necessary. Besides these emotional effects, complicatedness also leads to lower performance (Hacker, 1983) because of the reasons specified above.

IMPLICATIONS FOR SOFTWARE DESIGN

From the discussion so far, it follows, that software design should maximize control, optimize complexity and minimize complicatedness.

Maximization of Control

One can follow the outline of Table 1 to evaluate whether a program allows control. Word processing may serve as an example. Some word processing programs use "masks" for form letters, in which the curser moves to a particular position, in which the address of a number has to be inserted. Often, the sequence and even the timing of these masks is predetermined. This is a case of little control. Even the task content is often predetermined, as well (e.g., there is a preprogrammed dictionary of paragraphs that is often used in letters of a particular industry). Plans may be related to menus. The following questions exist: Can the typists change the sequence and content of the menus (e.g., are they able to turn the menus off, or can they develop their own menus, etc.). Decision possibilities with regard to feedback may be more difficult to program into the software. But it should be possible for a typist to design certain signals themselves. A typist may, for example, often forget to use a certain command. He or she might want to program a reminder into the system (e.g., reminding to save something). If these reminders were to be included in all programs, they would be patronizing and would often disturb the train of thought and action. If the users design their own reminders, they are useful tools (and can, of course, be changed again in case they are not needed any more). The users might also want to change the sequence of the signals. And finally, users might want to determine, individually, at what point they want to get signals. For example, it has been suggested that feedback should be given if response times of the computer are long (Shneiderman, 1980) -- people may differ here and it is useful to be able to choose at what point they would like to get feedback on response times. Control over conditions may mean that users are able to turn off sounds or the blinking of the curser, as well as using different types of keyboards or programming the keyboards or certain function keys. Control over conditions of work also means, of course, that users have an influence on which system is introduced and how it is introduced, which help systems (with whom as helper) they get, whether their work can be interrupted by commands from the outside, etc.

The issue of control becomes particularly important with the introduction of "intelligent" software. When, for example, an expert system "uses" the expert only as a helper of its program instead of the expert using the expert system, control is taken away.

It may not always be possible to give control to the user in every one of the cells of Table 1. But as the examples show, it is useful to go through the cells and determine to which extent one has been able to increase users' control. Control often implies that the user is able to change (even program) parts of the software. Two points are important here: First, the user should not be forced into having to change but should have the *option* to change (this is particularly true for the novice user). Thus, the system must provide a useful default option for each of the above mentioned examples. These default options should emphasize the development of a mental model and skills (the internal prerequisites of the experience of control). That means, that here explanations on the system functions should be given. Second, it is neither useful nor desirable that every user should become a programmer. People typically use the computer as tool to accomplish some other task (e.g. producing a text). Demanding that the user acquires complicated skills to change the work place, changes the focus from the real task to the tool. Rather, changing the program should be easy and computer assisted. Changing the system should use similar strategies as in "teaching" some robots. A robot is told by example what particular movements have to be done in which sequence. One button tells the robot that this is the example that is to be done from now on. Rather than having complicated strategies of telling what kind of mask a person wants to have, or in what sequence he wants to have programs loaded, a general "example- button" should be developed that automatically stores that particular strategy that the user prefers.

One of the more exciting developments in human-computer interaction has been the concept of direct manipulation (Altmann, 1987, Hutchins, Hollan & Norman, 1986, Shneiderman, 1983). With this design users see immediately whether an action led to the goal and the system is functional and transparent. Of particular importance are the chances to control the object and task directly (Hutchins et al., 1986). Therefore the operator feels in control.

Another development in software design is related to the concept of control: the concept of management of trouble by Brown & Newman

(1985). They suggest that the design should not be oriented towards decreasing any chances for errors because many design problems, tasks that a particular program is being used for, and personal approaches cannot be anticipated. Rather, it should make the errors controllable, i.e. it should make them manageable. Making the correcting of errors and the repairing of the effects easy and reducing the risks involved are more important design requirements than error reduction (cf. also Frese & Peters, 1987).

The internal prerequisites of learning a mental model and skills are, of course, also related to control. They will not be discussed in details here. However, it is interesting to note that control over the process of learning enhances training processes, as well (Frese et al., 1987).

It needs to be clear that having control also implies that the user is actively involved in the process of introducing new technology (e.g. a new computer system). The general goal is that the user *is able to design his or her work place* -- a principle that has long been discussed in industrial psychology as desirable (Ulich, 1978) but which has become a viable alternative with the advent of personal computers.

Optimization of Complexity

A very low level of complexity leads to boredom, a very high level to overload and to the frustration of not being able to achieve one's goals. Furthermore, people choose problems of moderate complexity. This means that the complexity of the software should not be reduced to zero. This should also not be done because it would reduce control as well (since both complexity and control hinge on decisions). However, that part of complexity that is difficult to control (complicatedness) should, of course, be reduced to zero.

Minimization of Complicatedness

Complicatedness should be minimized because it is a nuisance and it impedes performance. If complexity is not functional, intransparent, unpredictable, if there are no decision possibilities and little material to develop mental models and skills from, and if there is no social and technological adequacy, it turns into complicatedness. One way to reduce complicatedness is by reducing overall complexity (decision necessities). It is argued here that this would not be a good strategy because it leads to boredom, lower well-being and lower performance (cf. Hacker this volume). The better strategy is to increase functionality, transparency, predictability and the internal and external prerequisites of control experiences.

CONCLUSION

When developing software, the designer has to make choices. Most often these choices are not of the sort 'All arguments speak for this procedure' but rather of the type 'Some arguments speak for this route, but other arguments speak against it'. In short, the designer has to make choices under trade-off conditions. Since this is so, it is important to develop explicit rules of thumb on what alternatives are more important or less important. These rules of thumb should be made theoretically plausible so that they can be tested empirically.

In this article I have argued for the importance of experienced control and have tried to develop a conceptualization of control, complexity and complicatedness. I have argued for the maximization of control, optimization of complexity and minimization of complicatedness.

This is then the answer to the contradictory suggestions of the job enrichment literature (increase control!) and of software ergonomists (make it as easy as possible!): Too little complexity is boring and leads to an atrophy of mental capacities (thinking creatively) as well as to a reduction in performance. Moreover, if there is no complexity, there is usually no control either (because there are no decisions). If complexity and control do not go together, this leads to the experience of complicatedness. It is complicatedness, that the software ergonomists want to reduce to achieve user friendly systems. And here I agree, of course. However, sometimes software ergonomists may go a little far and reduce the complexity and control as well. Therefore the warning of the job enrichment people has to be heeded that a job renders an adequate amount of complexity and control.

There is one set of design decisions that our formulation is particularly critical towards: Reducing both complexity and control. For example, software that reduces the complexity level automatically when the user makes mistakes, reduces control and transparency over the system and should be avoided.

The emphasis on control is not entirely new to the field of software psychology (Cheriton, 1976, Shneiderman, 1980, Turner & Karasek, 1984, DIN- Entwurf, 1984). However, the relations between control and complexity and complicatedness have not been spelled out before. I do not think that my proposal answers all the questions. It is necessary to develop the concepts of complexity and complicatedness a little more formally than has been done in this paper. But I do think, that these three concepts are a starting point to analyze the work situation and software systems used in the work place.

The bulk of the arguments used in this article has not been from human computer interaction literature but from traditional (European) industrial psychology. This is not surprising. The question of good design of the work place has been the prime issue in industrial psychology for quite some time. The problems have not changed completely but are only accentuated differently when talking about software design (cf. Hacker in this volume). It is important to keep in mind that software design -- although becoming more important -- is only one aspect of job design. Therefore, good job design means to incorporate concepts of good software design into a complete work place -- hopefully a work place that allows a maximum of control.

FOOTNOTES

1) Acknowledgment: This paper is based on a first draft that was written while I visited the Center for Human Information Processing, Institute for Cognitive Science at the University of California, San Diego. Particularly Don Gentner, Don Norman, Dave Owen, Paul Smolensky and Judith Stewart have influenced my thinking about human-computer interaction. The visit was made possible by a travel grant from the Deutsche Forschungsgemeinschaft (No FR 638/2-1) which is gratefully acknowledged. I am also grateful to Siegfried Greif and Helmut Peters for their critique of earlier versions of this article.

2) Complexity and control are very often related. Semmer (1984) has measured both on two levels: by observers and by subjects' responses on a questionnaire. The correlations between the two variables were .70 and .43 on the respective levels. Karasek (1979) has even combined control and complexity into one index. Additionally the literature on the job restructering, control and complexity are seen as two sides of a coin: One should increase complexity and control in order to induce job satisfaction and development of the person (Ulich, Groskurth & Bruggemann, 1973, Hacker, 1985, Hackman, 1977). But these empirical correlations do not necessary imply that complexity and control are conceptually the same.

REFERENCES

Altmann, A. (in press). Direkte Manipulation: Empirische Befunde zum Einfluß des Benutzeroberflächen-Designs auf die Erlernbarkeit von Textsystemen. *Zeitschrift für Arbeits- und Organisations-psychologie.*

Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, Mass.: Harvard University Press.

Benbasat, I., & Wand, Y. (1984). A structured approach to designing human-computer dialogues. *International Journal of Man-Machine Studies, 21*, 105 - 126.

Boucsein, W., Greif, S., & Wittekamp, J. (1984). Systemresponsezeiten als Belastungsfaktor bei Bildschirm- Dialogtätigkeiten. *Zeitschrift für Arbeitswissenschaft, 38*, 113 - 121.

Brown, A. L. (in press). Metacognition, executive control, self- regulation, and other even more mysterious mechanisms. In F. E. Weinert & R. W. Kluwe (Eds.), *Metognition, motivation, and understanding.* Hillsdale, N.J.: Erlbaum.

Brown, J. S., & Newman, S. E. (1985). Issues in cognitive and social ergonomics: From our house to Bauhaus. *Human-Computer Interaction, 1*, 359 - 391.

Cakir, A. (1981). Belastung und Beanspruchung bei Bildschirmtätigkeiten. In M. Frese (Ed.), *Stress im Büro* (pp. 46 - 71). Bern: Huber.

Caplan, R. D., Cobb, S., French, J. R. P. (jr.), van Harrison, R., & Pinneau, S. R. (jr.) (1975). *Job demands and worker health.* Washinghton: NIOSH.

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

Carroll, J. M., & Thomas, J. C. (1982). Metaphor and the cognitive representation of computing systems. *IEEE Transactions on Systems, Man, and Cybernetics, 12*, 107 - 116.

Chin, D. N. (1986). User modeling in UC, the UNIX consultant (*Proceedings of the CHI'86 Conference on human factors in computing systems*). Boston: 24 - 28.

DIN 66 234 (1984). Normenausschuß Informationsverarbeitungssysteme (NI) (1984 als Entwurf verabschiedet). *Bildschirmarbeitsplätze. Grundsätze der Dialoggestaltung.* Deutsches Institut für Normung e. V.

DiSessa, A. A. (1986). Models of computation. In D. A. Norman & S. W. Draper (Eds.), *User centered systems design.* Hillsdale: Erlbaum.

Dörner, D. (1976). *Problemlösen als Informationsverbeitung.* Stuttgart: Kohlhammer.

Dörner, D. (1986). Heuristisches Wissen beim Lösen einer einfachen Steuerungsaufgabe (*35. Kongreß der Deutschen Gesellschaft für Psychologie*). Heidelberg: 1986.

Dunckel, H. (1985). *Mehrfachbelastungen am Arbeitsplatz und psychosoziale Gesundheit.* Frankfurt: Lang.

Emery, F., & Thorsrud, E. (1976). *Democracy at work: The report of the Norwegian industrial democracy program.* Leiden: Nijhoff.

Fitter, M., & Sime, M. (1980). Creating responsive computers: Responsibility and shares decision-making. In H. T. Smith & T. R. G. Green (Eds.), *Human interaction with computers* (pp. 39 - 65). London, New York.

Frese, M. (1978). Partialisierte Handlung und Kontrolle: Zwei Themen der industriellen Psychopathologie. In M. Frese, S. Greif, & N. Semmer (Eds.), *Industrielle Psychopathologie* (pp. 159 - 183). Bern: Huber.

Frese, M. (1982). Occupational socialization and psychological development: An underemphasized research perspective in industrial psychology. *Journal of Occupational Psychology, 55*, 209 - 224.

Frese, M. (1984). Transitions in jobs, occupational socialization and strain. In V. Allen & E. V. D. Vliert (Eds.), *Role transitions: Explorations and explanations* (pp. 239 - 253). New York: Plenum Press.

Frese, M., Albrecht, K., Altmann, A., Lang, J., Papstein, P. v., Peyerl, R., Prümper, J., Schulte-Göcking, H., Wankmüller, I., & Wendel, R. (1986). *The effects of an active development of the mental model in the training process: Experimental results on a word processing system.* Manuscript.

Frese, M., & Peters, H. (1987). *Zur Fehlerbehandlung in der Software-Ergonomie: Theoretische und praktische Überlegungen.* München: Manuscript.

Frese, M., & Sabini, J. (Eds.) (1985). *Goal directed behavior: The concept of action in psychology.* Hillsdale: Erlbaum.

Frese, M., Saupe, R., & Semmer, N. (1981). *Stress am Arbeitsplatz von Schreibkräften. Vergleich zweier Stichproben.* In M. Frese (Ed.), Stress im Büro. Bern: Huber.

Frese, M., Stewart, J., & Hannover, B. (1987). Goal- orientation and planfulness: Action styles as personality concepts. *Journal of Personality and Social Psychology, 52* (6).

Gardell, B. (1971). Technology, alienation and mental health in the modern industrial environment. In L. Levi (Ed.), *Society, stress and disease.* London: Oxford Univ. Press.

Gasser, L. (1986). The integration of computing and routine work. *ACM Transactions on Office Information Systems, 4*, 205 - 225.

Glass, D. C., & Singer, J. E. (1972). *Experiments on noise and social stressors.* New York: Academic.

Gleitman, H. (1985). Some trends in the study of cognition. In S. Koch & D. E. Leary (Eds.), *A century of psychology as science: Retrospections and assessments.* New York: McGraw-Hill.

Hacker, W. (1978). *Allgemeine Arbeits- und Ingenieurpsychologie* (2nd ed.). Bern: Huber.

Hacker, W. (1983). Psychische Beanspruchung bei Text- und Datenverarbeitungstätigkeiten an Bildschirmgeräten: Ermittlung und Gestaltung. *Zeitschrift für Psychologie, Supplement 5*, 24 - 41.

Hacker, W. (1985). Activity: A fruitful concept in industrial psychology. In M. Frese & J. Sabini (Eds.), *Goal directed behavior: The concept of action in psychology* (pp. 262 - 284). Hillsdale, N.J., London: Erlbaum.

Hackman, J. R. (1977). Work design. In J. R. Hackman & J. L. Suttle (Eds.), Improving life at work. *Behavioral science approaches to organizational change* (pp. 96 -). Santa Monica, California: Goodyear Publishing Company, Inc. (162)

Heckhausen, H. (1980). *Motivation und Handeln.* Berlin: Springer.

Hutchins, E., Hollan, J. D., & Norman, D. A. (1986). Direct manipulation interfaces. In D. A. Norman & S. W. Draper (Eds.), *User centered system design.* Hillsdale: Erlbaum.

Kahn, R.L. (1974). Conflict, ambiguity, and overload: Three elements in job stress. In A. McLean (Ed.), *Occupational stress* (pp. 47-61). Springfield, Ill.: Thomas.

Karasek, R. A. (1979). Job demands, job decision latitude and mental strain: Implications for job redesign. *Administrative Science Quarterly, 24*, 285 - 308.

Karasek, R. A., Baker, D., Marxner, F., Ahlbom, A., & Theorell, T. (1981). Job design latitude, job demands, and cardiovascular disease: A prospective study of Swedish men. *American Journal of Public Health, 71*, 634 - 705.

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

Kohn, M. L., & Schooler, C. (1982). The reciprocal effects of the substantive complexity of work and intellectual flexibility: A longitudinal assessment. *American Journal of Sociology, 84*, 24 - 52.

Kornhauser, A. (1965). Mental health of the industrial worker. New York: Wiley.

Maass, S. (1983). Why systems transparency? In T. R. G. Green, S. J. Payne, & G. C. van der Veer (Eds.), *The psychology of computer use* (pp. 19 - 28). London: Academic Press.

Miller, S. (1981). Predictability and human stress: Toward a clarification of evidence and theory. In L. Berkowitz (Ed.), *Advances in experimental social psyschology*, Vol. 14. (pp. 203- 256). New York: Academic.

Norman, D. A. (1983). Some observations on mental models. In D. Gentner & A. L. Stevens (Eds.), *Mental models.* Hillsdale: Erlbaum.

Norman, D. A. (1986). Cognitive engineering. In D. A. Norman & S. W. Draper (Eds.), *User centered system design.* Hillsdale: Erlbaum.

Österreich, R. (1981). *Handlungsregulation und Kontrolle.* München: Urban & Schwarzenberg.

Osterloh, M. (1983). *Handlungsspielräume und Informationsverarbeitung.* Bern: Huber.

Polson, P. G., & Kieras, D. E. (1985). A quantitative model of the learning and performance of text editing knowledge (*Proceedings of the CHI'85 conference on human factors in computing systems*). San Francisco: 207 - 212.

Polson, P. G., Muncher, E., & Engelbeck, G. (1986). A test of a common elements theory of transfer (*Proceedings of the CHI'86 Conference on human factors in computing systems*). Boston, 78 - 83.

Rohr, G., & Tauber, M. (1975). Virtual objects and virtual places how people comprehend their tasks using complex application software (*Macinter-Workshop on "Knowledge and visual information representation"*). Stuttgart: Talk.

Rouse, W. B., & Morris, N. M. (1986). On looking into the black box: Prospects and limits in the search for mental models. *Psychological Bulletin, 100*, 349 - 363.

Schardt, L. P., & Knepel, W. (1981). Psychische Beanspruchungen kaufmännischer Angestellter bei computergestützter Sachbearbeitung. In M. Frese (Ed.), *Stress im Büro* (pp. 125 - 158). Bern: Huber.

Seligman, M. E. P. (1975). *Helplessness: On depression, development and death.* San Francisco: Freeman.

Semmer, N. (1982). Stress at work, stress in private life and psychological well-being. In W. Bachmann & I. Udris (Eds.), *Mental load and stress in activity: European approaches* (pp. 42 - 55). Amsterdam: Elsevier.

Semmer, N. (1984). *Streßbezogene Tätigkeitsanalyse: Psychologische Untersuchungen zur Analyse von Streß am Arbeitsplatz.* Weinheim: Beltz.

Semmer, N., & Frese, M. (1985). Action theory in clinical psychology. In M. Frese & J. Sabini (Eds.), *Goal directed behavior: The concept of action in psychology* (pp. 296 - 310). Hillsdale: Erlbaum.

Shackel, B. (1985). Ergonomics on information technology in Europe - a review. *Behaviour and Information Technology, 4*, 263 - 287.

Shneiderman, B. (1980). *Software psychology.* Cambridge Massachusetts: Winthrop Publishers.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *Computer, 16*, 57 - 69.

Smith, M. J., Cohen, B. G. F., Stammerjohn, L. W. (jr.), & Happ, A. (1981). An investigation of health complaints and job stress in video display operations. *Human Factors, 23*, 387 - 400.

Turner, J. A., & Karasek, R. A. (1984). Software ergonomics: Effects of computer application design parameters on operator task performance and health. *Ergonomics, 27*, 663 - 690.

Ulich, E. (1978). Über das Prinzip der differentiellen Arbeitsgestaltung. *Industrielle Organisation, 47*, 566-568.

Ulich, E., Groskurth, P., & Bruggemann, A. (1973). *Neue Formen der Arbeitsgestaltung. Möglichkeiten und Probleme einer Verbesserung der Qualität des Arbeitslebens*. Frankfurt: Europ. Verlagsanstalt.

Volpert, W. (1974). *Handlungsstrukturanalyse als Beitrag zur Qualifikationsforschung*. Köln: Pahl-Rugenstein.