Swedish National Defence Research Institute
Department 1
S-1O4 50 Stockholm 80
Sweden

FOA 1 Report
C10029-M3(E5,H9)
July 1975
Second revised edition: July 1978
Third edition: November 1996

INTERACTIVE SOFTWARE FOR HUMANS

By Jacob Palme

Abstract:

Which are the human needs that are affected by the use of computers? How will computers affect the humans using them? What kinds of communications between humans will the computers cause? How can computer systems be designed to better satisfy human needs? What design principles should be used for such systems? What kinds of human-machine interaction will better satisfy human needs?

This paper discusses these problems and presents various methods. Computer driven, command driven and natural language interaction is discussed.

Search Key: Computer, Data processing, Human, Psychology, Sociology, Linguistics, Language, Communication, Interaction, Conversation, Politics, Data Base, EMISARI, people.

Swedish Abstract:

Vilka mänskliga behov påverkas av datorerna? Hur kommer datorerna att påverka de människor som använder dem? Vilka slag av kommunikationer mellan människor kommer datorerna att medföra? Hur kan man konstruera datorsystem, 5å att de bättre tillfredsställer mänskliga behov?

Denna rapport diskuterar dessa problem och beskriver olika metoder. Bl.a. diskuteras datorstyrd, kommandostyrd och naturligt språkkommunikation mellan människa och dator.

Foreword To The Third Edition

This report was originally published more than twenty years ago. The development of new techniques for human-computer interaction mean that some of the text in the book is somewhat obsolete. The basic ideas about human needs and computers, however, are still valid.

I have not changed the original text in this third edition, but I have used a smaller font for sections of less importance, and larger font for sections of more importance, in order to make it easier for readers to find the important parts of the book.

HOW THE TEXT OF THIS REPORT WAS PREPARED

The text of this report has been extended and modified many times. A number of people have read it and commented on it (Lars Enderin, Arne Grip, Hans Karlgren, Kalle Mäkilä, Mats Ohlin, Per Svensson, Jan Wirstad). Thank you!

In spite of this, no typist has had to retype the text several times. Instead, the text was stored in a computer, and an automatic text editor and page formatter was used to make changes and additions and still producing nice copy.

This is an example of how a computer can abolish dull and tedious work (retyping almost the same text many times). That is how computers should be used!

TABLE OF CONTENTS

1. INTRODUCTION

The price/performance ratio of computers is rapidly going down. The lower prices mean that people are finding the use of computers feasible for more and more applications. More and more people are becoming involved with computers.

Are we moving towards a "computer age" in which computers have taken over much of the communication now handled by other means? A "computer age" in which most humans will spend a large portion of their time communicating with computers? Or will the trends stop?

The future is not shaped by chance, it is shaped to a large extent by human actions. Will humans want or accept more computers? Should they?

The purpose of this paper is to present certain problems which occur when humans interact with computers. I will then present techniques for man-machine interaction, and discuss if they can solve these problems.

2. HUMAN NEEDS

Computers are for the good of humans, not humans for the good of computers. A good starting point in discussing man-machine interaction is therefore the human. Which are those human needs, whose fulfillment is influenced by computers? Here is an incomplete list of them:

- ACCEPTANCE: A human needs to be accepted by his peers. people should have time to listen to him, when he has something to say. They should show respect for him as an individual. (See Harris 1969.)
- TWO-SIDED CONFLICT-SOLVING: When a problem occurs, people should try to find a solution which reasonably well satisfies the needs of all people involved. One person or one side should not impose its solution on the other side. An important initial phase of two-sided problem-solving is that both sides listen to each other and try to understand the other persons needs and ideas. (See Gordon 1970 and Gordon 1974.)
- PERSONAL DEVELOPMENT: A person should have the right and the opportunity of developing himself, of learning more, of improving his skills.
- ABILITY TO MODIFY HIS ENVIRONMENT: A person should have the means of changing his environment, of improving the environment when it is bad for him. A recent study has shown that a feeling of helplessness, that is, a feeling of not being able to solve problems by influencing one's environment, is an important cause of human depressions. (See Seligman 1975.)

Sociologists talk about AUTHORITARIAN and DEMOCRATIC social systems. The authoritarian system is based on rigid lines of commands, people are expected to do what they are told and not ask questions. In the democratic system, communication is more two-way: There is no boss who dictates his wishes without listening to and being influenced by the people. The democratic system will easier give solutions which are satisfactory to more people, and it will better satisfy the human needs listed above.

3. THERE IS A HUMAN BEHIND THE COMPUTER

(This chapter is largely influenced by Grip 1974.)

Compare two systems for doing the same job, one with and one without computers. Humans play an important part in both systems. But in the computer system certain functions are handled by computer programs.

3.1 THE PROCESS OF COMPUTERIZING A JOB

To understand the effects of this one must look at the process of computerizing a job. This process begins with the recognition that computers might be advantageous. Special experts are given the job of finding out if computers should be used, and if so how.

These experts usually have a technical education. They have been taught about computers and systems analysis, but not so much about humans and human relations.

This alone means that there is a large risk that they will design a system solution which does not satisfy human needs.

The experts will analyze how the job is done without computers. They will make a detailed plan of how the job should be done with the help of computers. This plan often means certain changes in the previous procedures to make them more suitable for computers.

Such a plan will include a detailed chart of the flow of information in the system: Which information is necessary as input to certain subtasks, and which information is produced by the subtask. Which tasks should be performed by humans, and which tasks can better be handled by a computer program?

The goal of the plan is to do the job cheaply, efficiently and well. Often, the specification of the job will be changed to include extra benefits, like more statistics, which at little extra cost can be produced when the job is computerized.

The plan is then presented to the management, which has to judge if the plan will agree with their long-range goals. They will check the plan against goals such as profit, product quality, public image, and sometimes also the well-being of their employees. Naturally, the management tends to check the system especially against their own needs. Among these are the needs of the management to control the flow of information and the lines of control.

When the plan has been approved, the experts take over again. A group of people begin to implement the plan, that is buying the computer, writing the programs, teaching the employees their new tasks in the computerized system etc.

This implementation process often takes a long time. Especially computer programming is often time-consuming and expensive.

Finally comes the day, often years later, when the new system is started. There are troubles in changing over, and sometimes the system will never work as intended. In such cases the system may be scrapped, or people will do the computerized sub-tasks manually beside the computer.

The reason for such doubling of the work may be conservatism among the employees, but may also be because the computerized system does not do the work as well as needed. The computerized system may not provide certain information needed, or may take too long to get certain things done. Since the development of the computer system has taken several years, it may be partly or wholly obsolete.

People will easier accept and be satisfied with computer systems if they find that the system has benefits for them. If a person only has disadvantages from the system, but does not get anything back which he finds valuable, then he will probably not be satisfied with it. It is therefore important not to design systems with the only goal of providing information for the top people in an organization. The needs of the people at lower levels must also be taken into account. If each person who has contact with the computer system also can see direct personal benefits from the system, then he will better accept the system and it will work better.

If the system is successfully implemented and people do begin to use it, there are still problems. People will always complain about deficiencies in the computer system. It does not provide certain needed information in a certain form, or certain common tasks are too difficult to make with the computers, or the computer system is error-prone in certain ways, etc. etc.

This means that there must be people who change the computer programs. For large systems, there is a large group of computer programmers continually doing nothing except updating the system to satisfy new user requirements.

But this is a time-consuming process, and a change needed by a user may take months or years to realize, depending on its priority.

3.2 THE COMPUTER WILL DO WHAT IT IS PROGRAMMED TO DO

Those tasks of the new system which depend on the computer will only work in the way the computer is programmed. The previous system was probably a manual system, where humans performed directly many tasks now delegated to computer programs. In this manual system, there were ways of bypassing the normal routines. Questions could be answered, orders processed, short-cuts devised for special tasks, new ways devised for new problems.

The manual system could continually adjust itself to new situations or changes in the task. These adjustments were often made in a quick and flexible manner. They did not require months of work of highly trained computer specialists, they could be done in a few minutes by just a human talking to another human.

Note, however, that if the system before the computer was partially automized using less powerful machines than computers, e.g. ledger-poster-machines, then the computer may give more flexibility than the previous machines.

Why then does the computer system not provide the same flexibility as manual systems? The excuse always given by the computer people is: This was not foreseen from the beginning. If we had only known that you wanted to do this and this and this, then we could have made such a system. But you never said this when we asked you what the system was to do. And you accepted our plan of the new system. And now it is very difficult to put your changes in afterwards.

The basic fault seems thus to be that everything was not foreseen from the beginning. But no one can foresee all future needs from the beginning. It is usually not even possible to make a complete inventory of the needs today, or of all the functions provided by the manual system which is to be computerized.

The fault is rather in the whole design process. The typical design process, as described above, involves studying the existing system and creating an ideal plan of how the system should work. Very often this ideal plan is imperfect in itself, especially if the future users of the system did not take part in the design process. Example: it may only optimize technical efficiency without taking human needs into consideration. Even if the plan is good, it still gives some people a static picture of the system at just one time.

The best design process is one which produces systems which will satisfy even those requirements which were not explicit when the system was specified. This is possible if the system is so designed that the users themselves can modify the system to do new or different things.

Most systems do not give the users that facility. The computer only works in the way it has been programmed. Everyone (systems designers, programmers, office workers) are forced by the computer to work according to the original plan. The short-cuts and flexible solutions in the previous manual system are often no longer possible.

One can view this as a communication process. The computer is a means of communicating the plan to the employees. By designing and writing the computer program in a certain way, the experts are communicating to the employees their view on how things should be done.

This communication process is characterized by two things:

- The communication is delayed. The computer was often programmed a long time before the employee receives the message.
- The communication is authoritarian. The employee usually has to do as the computer people say, since otherwise the computer will not work.

It is obvious that such a communication process will often not well satisfy the human needs specified in section 2. above.

Example: The National Agricultural Board wanted all forest surveyors to use one way of estimating the value of forests. They then developed a computer program to be used for forest value estimates. This program could only handle estimates made in the way preferred by the National Board. There was a lot of grumbling among the surveyors, but everyone was forced to do the estimation in one way. Thus, the computer was used, by the National Board, as a method of imposing their will upon the surveyors.

Example II: In Stockholm, there was for many years a marked shortage of apartments. A public body was set up to distribute the available apartments to those who had the largest need and had been queuing for the longest time.

In 1975, the situation was the opposite. There was too many apartments, and the tasks of this public body was changed into helping people find the best apartment among those available. However, the computer program used to distribute apartments did not fit this new task. The program stored a lot of information about income, number of children, time in the queue etc., which was needed for the previous task of distributing apartments to those with the greatest need. The program stored little information about personal requirements on the apartment, since little choice had been available.

This example shows how a computer program can make it difficult to adjust a system to new needs in a changing environment.

If one computer is doing the task of hundreds of people, then a centralized decision to change this task can be made just by changing the computer program. Such a centralized decision might often be more difficult to enforce in a manual system. A computer will therefore sometimes create a system where centralized decisions of changes are simpler to implement, but where local changes without centralized approval are more difficult to make.

3.3 THE COMPUTER AS THE LONG ARM OF THE CENSOR

One can understand this by comparing with a famous section in George Orwells novel "1984". That novel depicts a future society in which a totalitarian state watches everything people are doing to check that nothing forbidden is done or said.

A section of the book contains a discussion how to stop people thinking illegal thoughts. It is observed that our thoughts are governed by the language we use. Orwell suggests the invention of a new language, such that illegal thoughts cannot be formulated in that language:

"The purpose of Newspeak was not only to provide a medium of expression for the worldview and mental habits proper to the devotees of Ingsac, but to make all other modes of thought impossible."

The computer has, in fact, already provided an instrument for partly realizing Orwells prophecies. A computer can talk about nothing but the topics it has been programmed to handle. The program designer can thus decide what kind of messages should be allowed or disallowed. The computer gives him power to enforce his decisions on the people using the computer.

The more we let computers take over human communication processes, the more these processes can be closely guarded and censored by the program, which acts as the long arm of the program designer.

Some examples:

- Some of the input to the computer has to be phrased in a complex specialized language which only a few people know. In that way the computer can be used to decide who is allowed to use the computer. Ordinary people are not accepted.
- The management would find it embarrassing if certain facts were known. The computer is programmed so that those questions cannot be asked. The output from the computer is phrased in such a way that certain things are very difficult to read. I have for example a wage slip from an employer (not my own) who perhaps did not want his employees to find out how the wage had been calculated. The various sub-fields are tagged by integer codes, explained in small print on the back, so that it becomes very difficult for someone who is not a expert to read and interpret it.

An interesting fact is that even when there seems to be no premeditated intention of the systems to satisfy the people responsible for the computer rather than the ordinary users, the systems still tend to come out that way.

It is not very surprising that one thing is very easy to read from the wage slip. That is the net amount of money being paid out.

In this way the designer of a computer system can make it easy to get information which he wants people to have easy access to, but he can make it difficult or impossible to get information which he does not want people to have easy access to.

This is in some ways similar to the special languages which professionals (lawyers, medical doctors, clergymen etc.) of all times have used (See Hoare 1975). But the computer is a more powerful tool for enforcing a mode of communication. This mode of communication is also often much more restricted.

3.4 YOU ARE ALWAYS COMMUNICATING WITH A HUMAN

Every communication between a human and a computer is in reality a communication between humans:

- The computer system is constructed by humans.
- The computer programs were written by humans. They laid down the rules for how the computer should answer questions from a human user.
- When a human stores information in a computer, which is later transmitted to other humans using the system, this is also a delayed human-to-human communication process.

However, the human-to-human communication through computers is different in several ways from direct, face-to-face human conversations:

In a human-to-human face-to-face conversation, the participants can through an iterative process come to an understanding of the problem and each other's views, and then find a solution based on this mutual understanding.

- In a human-to-human face-to-face conversation, natural built-in human factors protect us from inhuman decisions. It has been said that it is easier to kill millions of humans by pushing a button than to kill one human face-to-face. The same principle applies also to less severe acts than killing.
- In a human-to-human face-to-face conversation, the facts about the people present are taken directly from each person himself. Mistakes and misunderstandings will then be less common. With a computer, facts are instead often taken from data bases which may be faulty or misleading through incompleteness.

  Constructed example: An employer is going to select someone for a task requiring a reliable person. The employer uses data from a computerized data base. Two people are equally merited, but one of them was on sick leave four weeks last year. (The computer does not contain the cause of the sick leave: an accident in the workshop.) The other person gets the job. And the next time someone is to be chosen for a difficult task, he will have had more experience with such tasks. This example shows that incomplete information from a computer can hit a person unjustly.

- In a human-to-human conversation, the listener is continuously informing the speaker if he understands and agrees, so that the speaker can adjust his language to the reactions of the listener. See further Chapanis 1975.

Communication between humans through a computer has some similarities with written communication through letters, forms or other documents. However, the computer can be made to govern the communication, e.g. by selecting what to tell whom. This ability of the computer is felt beneficial for those who can tell the computer how to govern the communication, but may be felt less acceptable for those who cannot modify the workings of the computer according to their needs.

Another difference between communication through computers and through written documents is the speed with which the user gets a response. If the response can be generated from the programs and data already in the computer, the user can get an immediate response from the computer just like in a vocal human-to-human communication.

3.5 SOME HUMANS PREFER COMPUTERS TO PEOPLE

Many people will sometimes find face-to-face communication with other people troublesome. They find certain communication situations emotionally trying or stressful. People might then prefer to communicate with other people through a computer.

Example I: An extreme example of people who do not accept normal, vocal communication with others are certain children who do not speak at all. Such children also often have violent seizures of temper. The cause of this is believed to be that these children have been frightened off human communication because of psychological problems with such communication, which the children could not solve.

In an experiment, described in Kent 1975, such children have been treated by letting them interact with a computer programmed to be very patient and very responsefull to the wishes of the children. The machine does not, for example, mind at all if the child asks it, a hundred times in succession, to show a running fire engine on the screen, together with suitable sound effects from a loudspeaker. The computer treatment was applied to twenty-five children, on whom all previous treatments (with human therapists) had failed. In a large majority of cases, the children improved much in their ability to communicate with humans after having taken the first steps together with a computer.

The probably reason for this result is that the computer presented a communication situation which did not have the frightening aspects, which human communication had to these children.

Example II: Many people find it difficult to speak up in a group of other people. When communicating through a computer, such people are often found to have valuable contributions to make which were previously unheard because they dared not say their opinion. The computer gave them the possibility to phrase their contribution carefully in a situation which was less stressful to them than a meeting with a group of other people.

Example III: Certain people with emotional contact problem find it difficult to establish working contacts with other humans. At the computer, they find solace in a machine which talks back to them in a way they can understand and master. (One can compare with people who prefer pornographic magazines to the real thing, for similar reasons.)

Example IV: A human who wants to or has to hurt another human may prefer to deliver the message through a computer. The computer is a machine which can be used to manipulate others, with the manipulator protected from certain unpleasant consequences which would occur in face-to-face encounters.

The use of computers has obviously both advantages and disadvantages from this viewpoint. The advantages is that certain people will at certain times prefer, and perform better in, the protected environment provided by the computer. The disadvantage is that they may sometimes flee to the computer when direct face-to-face communication would have been better. Example V: A person prefers to get a regular financial report from the computer because it gives the same report format computed in the same way reliably every time. Note however, that if the computer prepares its report from data which humans have given it, then these data may have hidden subjectivity, and the objectivity of the computer report may be misleading and therefore dangerous.

A commonly stated view is the fear of private personal information in computers. However, computers can be designed to protect private personal information better than manual systems, e.g. so that no one is every allowed to take out the private data except the person himself. Others would only get statistical tables of means etc. where single persons are not identifiable. In this case, people may prefer computers just because they protect data better than manual systems.

One might claim that some people, who prefer computers, would in the long run be better helped by psychotherapeutic treatment to overcome their contact problems, rather than with a machine to make it even easier to hide. However, such treatment is not always very effective, and the computer may be a more workable solution (Changing the environment rather than changing the people).

The wish to use a computer to avoid certain stressful human communication situations can sometimes also be viewed as a natural and responsible way of reacting to certain situations.

3.6 COMPARING COMPUTERS AND MASS COMMUNICATION

The increasing use of computers for human communications has thus obvious dangers compared to human face-to-face conversation. However, sometimes computers replace human communications which were previously transmitted by one-directional, written media like letters, pamphlets or mass communication (radio, TV, newspapers etc.). One of the differences between human-to-human communications through mass communication and computers are that the computers increase the possibility to selectively decide who is to receive or not receive a certain message.

This can be done in two ways:

- By programming the computer to check information about the individual (his occupation, his interests etc.) to decide if he is to receive a certain message.
- By letting the human who wants information get it by asking questions to a computer system.

In both these cases, the computer can give different information to different users. Mass communication media are much more crude. The computer is better at selectively distributing information to those who need it, or whom the sender needs to transmit it to.

If a computer system answers the questions which the user wants from it, then he will often be much more satisfied with the computer than with mass media.

If a computer system distributes information automatically, using selection criteria set by the receivers of the messages, then the receivers will also be more satisfied.

However, the computer system is often governed more by the sender of the messages than by the receivers. The sender decides what to store and who should be selected to get certain messages (e.g. advertisement material), the sender decides which questions the computer should answer and not answer.

If a sender decides not to send information to someone needing it, that person may feel neglected and may not be able to assert his rights. If many senders send too much information to someone, that person may feel overburdened with information he does not need and did not ask for.

In most cases, a sender-driven system for distributing information through computers is dangerous, while a receiver-driven system is beneficial.

Systems which are basically sender-driven can be improved by giving the receivers some right to ask for information not sent to them, and some right to stop information they do not want.

Some people have claimed that such receiver-driven computerized communication systems will be used by people to restrict their new impressions to a limited area of interest. Since different people do this in different ways, they will have less knowledge in common and therefore find it more difficult to communicate. Society will more be divided into groups with their own interests and ideas.

I personally feel that variation in society is beneficial, a powerful driving force towards testing and trying new ideas and new approaches.

It might also be possible to design computer systems which help people get away from confinement into restricted areas of interest. Note that conventional communication restricts an individual to rather coarse interest groups: the readers of a specialist magazine or the members of a specialized society. In non-computerized systems, the movement to another interest may be a large step. With computerized systems, a person can change his interest profile in a number of small steps, slowly and carefully moving towards new interest areas.

3.7 THE HUMANS INVOLVED WITH COMPUTERS

Different groups of people are involved with computer systems:

- Manufacturers and machine designers,
- Employees, whose tasks are changed or taken over by computers,
- Managers, who hope to get better efficiency with computers,
- Experts (System designers, programmers, repairers etc.), a group whose size increases rapidly with computerization,
- Outsiders, that is people outside the place where the computer is used, who are influenced in various ways by it.

The experts are a group which generally has high status and a rather varied, creative work. Because everyone who uses a computer system has to use it the way it is programmed, these experts also get power. For all these reasons they usually get good work satisfaction from computers. The fact that boring routine tasks are taken over by machines handled by such people will thus mean better personal satisfaction.

In some places, the process of computerization has moved much power within a place of work from the ordinary lines of management to the computer system developers. This has caused conflicts with the ordinary management, which feels bypassed by the computer people.

This does not of course mean that all computer experts are always satisfied with their jobs. The computer system may be so badly designed that not even the experts can get it to do what they want in reasonable ways. Sometimes, the managerial organization makes them less satisfied. For example, the organization may be such that the experts at a lower level are not able to see the meaning of their work, perhaps, but not necessarily, because the work really is meaningless or detrimental. In such a situation, the experts tend to try to find satisfaction in the intellectual challenge presented by their tasks. But do we want such experts? Groups who may be in trouble are the ordinary employees and the outsiders.

Some ordinary employees lose their jobs. In many cases, it is difficult to find new jobs for them, because the new skills are so different or because they have to move. They are therefore people who are in trouble.

Example: In newspaper production, manuscripts are more and more often stored in computers, and the typesetting is done automatically by the computer. This takes away some of the work of the typesetters. The typesetters will then require that they get the new jobs of handling the computer and inputting data to it. However, in many cases it is natural that a journalist or an advertisement receiver inputs the data directly, and then the old typesetters cannot take over that job unless they become journalists or advertisement receivers.

A computer is a very good machine for doing simple, routine tasks automatically, guided by its program. One would therefore expect that the computer would remove boring routine jobs and let people do jobs more suitable to humans. To some extent this is also true. But often the result is different. Because the computer systems are designed to do the task in just one way, the ordinary employees have less freedom than before. Also, if the computer takes over many steps in a manual process and leaves only one step for a human, this means that the job for this human will have less variation.

When jobs become too simple because of automation, a common way of treating this problem is to change the work plan so that each person handles a series of tasks. Instead of having separate people just keypunching data into the computer, the person who receives the data (e.g. customer orders) could both talk to the customer and input the data into the computer. In other cases, it is possible to let the data go automatically into the computer without any human intermediary. For example, a balance might transfer weight information directly into the computer.

The computer systems can be programmed to monitor everything that is going through them. They can monitor the speed and efficiency of every employee, and they are often programmed to do this. This continuous supervision is a closer control of the employees than with a human supervisor who sometimes goes away.

Example: At the Viking Askim rubber company in Norway, the workers discovered that their tools were connected to a computer, which received a signal for every operation made on the tool. A strike resulted, and the strike was resolved by an agreement that the company should not use the data from the tools to do any kind of individual performance measurement on the employees.

The managers will sometimes find that they too are restricted by the rigidness of computer systems, even though they have themselves approved of the specifications. The tasks are changing, and an old, large computer system may be a hindrance to changing the organization in accordance with a new situation.

If ordinary people in the street are asked their opinion about computers, they often express feelings of unease. They regard the computers as instruments for increasing centralized power, because computers have been used very much in this way. The computer has become a symbol of centralized power in a way which mean that people transfer their negative feelings about centralized power to negative feelings about computers. Can this public feeling towards computers be changed into a more positive feeling, if computer systems are developed and used in ways which people conceive as beneficial and useful?

3.8 EMPLOYEE PARTICIPATION IN THE DESIGN PROCESS

In Norway, experiments have been made with employee representatives in the design process. The Norwegian Trade Union Federation was very early aware of the risks for the employees with computers. They felt that computers, if governed solely by the management, could mean more power and control in the hands of the management and less influence for the employees. They started experiments with employee participation, contracted computer consultants of their own and they have today arrived at a written agreement with the Employer's Association Federation about the use of computers.

The agreement says that not only technical and economical, but also social considerations shall be taken in the design of computer systems. The employers must at a very early stage in the design process tell the representatives of the employees about their plans. Those employees who are directly concerned with the computer systems should also take part in the design work. The agreement specially states that information should be given to the employees in a language which they can understand.

The employees can, if they so want, appoint a special representant for computer problems. This representant shall have access to all documents about the computer system and take regular part of the design work. The representant shall, at the employers cost, get the necessary education about computers.

A local agreement must be reached about all use of computerized personnel files. If an agreement cannot be reached, the conflict should be moved to the national associations.

Such employee participation has many advantages:

- The employees learn enough about the computers to be able to put force behind their requirements.
- The designers are forced to listen to the employees, and this is an important way of teaching computer system designers more about how to design better systems.

The problems with employee participation are:

- The employees do not always know enough to be able to talk back to experts who say: "This is not possible."
- The employees may be compromised to accept systems which in practice are found to be bad. The designer can then say: Well, you accepted our specifications!

I personally believe that employee participation is very important. But I also think that it would be dangerous to believe that this will solve all or most of the problems.

An ideal computer system may not be one which satisfies all the requirements put by the users during the design process, but rather a system which is flexible enough to be able to adjust itself to the needs of the users on a day-to-day basis.

A special kind of use of computers is operational analysis models. These models are developed to simplify decisions by doing much of the routine computation necessary in certain decision-making. The models are developed in close cooperation with the decision-maker (e.g. the management) so that it suits their needs.

One should be aware that the employees are generally not aided if they just get access to such models done for the management. This is because these models are adjusted to the needs and views of the managers. They may even be dangerous, since they may mislead the employees into the false belief that there is no decision possible other than that which comes out of the computerized model. The true fact is that all models are simplified, usually in such a way that only certain outcomes are possible.

Example: A very common kind of such model is that used for investment computing. The model helps managers decided when to do investments, and to see which investments will be profitable. However, most such models do not take the influence of investments on employment into consideration. The fact that investments should be distributed over the years in such a way that sudden increases and decreases in the number of people employed can therefore not be taken into account. Obviously, such a model is dangerous to the interests of the employees.

3.9 COMPUTERS AND HUMAN NEEDS

If we look at the human needs listed in section 2 above, we thus find:

- ACCEPTANCE: Computers will often give people a feeling of insecurity. The computer system poses rigid requirements on its users, but the person who designed the system and invented these requirements is far away. The typical attitude of the designer of the system is that of an expert talking to laymen. He is transferring this attitude to the computer system, so that the typical attitude shown by the computer to its users is: I'm OK, you're not OK. (See Harris 1969.)

  Even when the computer is not OK, when the user knows much better, it still transmits this attitude which it has been given by its designers. The continuing supervision of its users performed by some computer systems for their managers also add to this situation. Thus, the computer will often cause situations where the human need for ACCEPTANCE is badly satisfied.

  Sometimes, computer system designers give people the feeling that their job is something a machine will take over in the near future. This may make the people feel unaccepted, if there is no other better job they can do instead.
- TWO-SIDED CONFLICT-SOLVING: A computer is often a tool for one side, the designer and his principal, to impose their view how things should be done. Such a computer will badly satisfy this human need.
- PERSONAL DEVELPMENT: In a manual system, a human usually has the ability to modify his way of handling things do them better. With the computer, this ability is often circumscribed by experts difficult to reach.
- ABILITY TO MODIFY HOS ENVIRONMENT: Since the ordinary user usually does not have the right or ability to change the programs of the computer, this human need is circumscribed. Computer programmers, however, often feel that computers are good just because the computers give them this ability of modify the environment.

3.10 CAN WE DESIGN BETTER COMPUTER SYSTEMS?

Can computer systems be designed in such a way that they will better satisfy human needs? I believe so. Here are some of the things we could do:

- Postpone the decision about computer or not until the understanding of the issues is strong enough to make such a decision.
- Teach the designers of computer systems much more about human needs and about how systems should be designed to satisfy them better.

- Let the employees and the users take part in the design process, not only the designers and the managers.
- Teach ordinary people more about what computer systems can do, so that they ask for more and do not accept bad systems.
- Always remember that the goal is to get the computers to help people. Each computer task should be question from this viewpoint: Which people are helped by this and how? Do they feel so themselves? Are other people negatively affected, and if so how?
- Create good computer systems, publicly available, so that people are shown what can be done. They will then not as easily accept other less good systems. An example of such a good public system would be a system with data bases containing information people often need, and with terminals at public libraries where anyone can use them.
- The way to design good systems is not to design them so that they can only do things the way the designer plans. Instead, the users of the system should have the ability to make the system do things the way they want. The user should be in command of the system, not the system in command of him. This will be discussed further in the next section of this paper.

4. FROM HUMAN NEEDS TO TECHNICAL REQUIREMENTS

This chapter will discuss various ways of arranging the man-machine interface.

The main conclusion of the previous chapter was that a computer system will be more acceptable to humans if the system gives power to the humans who are working with it. If the humans who are working with a computer have power over the system, can get the system to do what they want, if the system and the computer is a useful tool helping them, then they will find the computer system acceptable.

If, on the other hand, they are guided through the computer in an authoritarian way, restricting their freedom and their possibility to solve their problems in a simple, natural and flexible way, then the computer system will be an obstacle to them.

How then can we design computer systems to provide power and freedom to the people who get involved with them?

4.1 ALL DATA EASILY AVAILABLE

The person who is using the computer should have available to him all the information which he needs, and which is stored in the computer. The important thing is here what his needs are regarded to be. Often, the system designer has an "ideal" model of the tasks performed by the people using the computer, and each of them is then given access only to the data needed to perform this "ideal" task.

However, this does not take into account the fact that in special circumstances the user very often wants to do something different from the "ideal" task. He should then have the facilities to get the information he needs. The best way to give the user this freedom is to

a) Store all data in a general-purpose data base handling system,

b) Let every user have access to a general-purpose inquiry language which can get all the information in the data base handling system.

To avoid misunderstanding, I do not mean that all data base accesses always must go through user queries in the general-purpose inquiry language. But this facility should be available when needed.

A conflict may arise with secrecy requirements for certain data, e.g. military secrets or personal data.

An important political question in the future will be what information should be available to whom. Is it necessary with so much commercial secrets, or do they hinder progress? Will computers be used to make more facts available to more people, or will they be used to protect information and only make it available to a small elite?

A computer can be a very powerful tool for protecting information. But a general-purpose inquiry handler will be less useful and meaningful if the main goals are protection rather than openness.

Very important is that people designing data base systems are aware of this conflict. It is so easy to use the secrecy requirements as a pretext for designing a system which forces all users to work as the system designer wants them to and for dismissing the human needs of the people who are going to use the system.

A common complaint against computers is that they produce to much information. A human mind does not have the facility to grasp and handle large amounts of information. A human therefore wants to have only the most important data presented to him. He wants data ready treated for his needs. If, for example, the user asks for the number of accounts larger than 2000, he does not want to get a list of all the accounts, he only wants a number. Important is that the query language is flexible enough to allow the user to select exactly what he wants.

With a conversational system, it is especially important that the user does not have to wait for long printouts of things he does not want. Every message to the user should be short. With a display terminal, the message should be small enough to fit into one screen. This requires a flexible query language to allow the user to state exactly what he wants. If a user asks a question which requires lengthy output, he should be told so and asked what to do about it before the long answer is output. is just the kind of power over the computer which is good for the human.

4.2 THE COMPUTER SHOULD ADJUST ITSELF TO THE USER

```
Human        |
satisfaction |
and          |
performance  |                  ooo
             |                o  |  o
             |               o   |   o
             |              o    |    o
             |             o     |     o
             |           o       |       o
             |         o |       |       | o
             |       o   |       |       |   o
             |    o      |       |       |      o
             | o         |       |       |         o
             +-----------+-------+-------+----------------- Requirements
                         a       b       c                  put on the
                                                            human
```

The curve above illustrates how the satisfaction and performance of a human varies with the requirements put on him. If too much is demanded of him (point c on the curve) then he will be overstressed, and his satisfaction and performance will be lowered.

If, on the other hand, too little is demanded from him (point a on the curve), then he will find his task dull and tedious. He will find it difficult to concentrate on his job, and his satisfaction and performance will again be lowered.

He will be most satisfied and perform at best if the requirements on him are well adjusted to his abilities and ambitions (point b on the curve).

Since people are different, the first conclusion from this is that the same mode of interaction is not the best for every user.

For example, an inexperienced user may prefer a computer which guides him along with explanations and explanatory questions. A very experienced user may on the other hand prefer a very powerful and concise command language, in which much can be said to the computer with few keystrokes.

But even for one single person, his requirements vary with time. His requirements are different when he begins to use a computer and when he has much experience with it. And changes in his tasks will also influence what computer behavior is best for him.

For example, a special work situation one week may require him to input an almost identical series of inputs to the computer a large number of times. He then needs a facility to shorten this special series of inputs just during that week.

The repetition of the same trivial series of commands many times can become tedious and dull for a human. Therefore the computer should take over such repetitions when needed by the user. The ability to turn such repetitious work over to the computer

Because a computer can be programmed, the computer can be made to adjust itself to different user needs much easier than many other machines. This advantage of the computer compared to other hardware should be used.

When two humans talk to each other, they will automatically adjust their language so that the other understands. (One of many cases where humans are more automatic than machines!) A computer can be programmed to have at least some of this ability.

4.3 A HUMAN HAS THE RIGHT TO CHANGE HIS MIND

Linguists sometimes make transcripts of spoken human language. These transcripts are very interesting to read. They contain a large number of linguistic errors. The speaker stops in the middle of a sentence, restarts something else, leaves out important parts of his phrases, makes a large number of trivial errors which he immediately corrects. Obviously, this way of talking is natural for a human. People listening are usually not irritated either, they have the ability to understand such incomplete and corrected language. Since this is a natural mode of using language for a human, a good computer should also be programmed to allow it. Thus, a human at a computer terminal should have the ability to correct things he has previously put into the computer.

Ideally, he should be allowed to go back as far as he wants in the conversation, correct only that what he wants to change, and then continue where he stopped. He should not be forced to input again a number of already correctly input data just because he wants to change one previously input item.

Certain computer programs work in such a way that the user is led through a program-defined path of questions and stages of execution, with none or very restricted possibilities for the user to change his mind about previous answers to the questions. Such systems are not good.

Even when data has already been stored in a data base, there is still often a need to correct it. Example: A customer may call to change or cancel a previous order.

The problem here is that there may be other data dependent on the corrected item. Example: First a person got too much pay. Then this was corrected, but too much tax was deducted, based on the previous faulty wages, so that he now got too little money.

4.4 HUMANS AND MACHINES MAKE ERRORS

To be able to work at all, a lot of time in every large computer system is spent on error-checking at different interfaces: User - machine, program - language system, language system - operating system etc.

If an error anyway has to be corrected, it is much better to correct it as early as possible.

Important is that the error is explained to the user in a language relating to his terms of reference. If he by mistake tells the computer that the name of someone is "45', then the computer should answer "I do not understand numerical names". The computer should not answer, e.g., "Illegal memory reference" or "Pushdown " list overflow" or "Array bounds error .

The computer should answer politely, especially in cases where the computer may be in error rather than the user. If the computer expects a number and gets the input "twenty", then it should not be programmed to answer "ERROR: Input was not number".

The person who writes a computer program tries to envisage what kinds of correct and incorrect answers his program will receive. But no person is ever able to envisage all kinds of correct and incorrect answers people will actually give to the program in field use.

A good way of finding out is to write the program in such a way that it saves all answers received in a file. In my experience, when the program has been tested by less than fifty different human users, then this file will not be certain to contain enough data to predict almost all kinds of answers which people will try to give to the computer.

Only when the program has been corrected based on the data in such a file can it be called acceptably responsive to human behavior.

If the error checking in the computer system is bad, then a small mistake by a human at a terminal can cause much trouble. Obviously this risk puts stress on the human. Therefore, good, early and immediate error checking by the computer will reduce stress.

Example: Cashiers in post offices in Sweden have a responsible task. They are also personally responsible for losses, which they by mistake may cause the post office. A lost receipt can cost the cashier thousands of kronor. This risk makes the work of the cashiers stressing. In the last year, the post offices have begun using cashier computer terminals. Inger Evenfelt, who has been working in the post since 1948, says "This is the best that has been happening in 25 years." She thinks of the error and validity checks in the computer program, which helps her avoid her own mistakes. This is an example of a property of a computer program which often makes people like using it. In direct human-to-human communications, the people involved will adjust their error checking to the special needs of a special day and to the special characteristics of the other people involved.

Example I: When inputting data about individual persons, other checks may be appropriate then when inputting data about enterprises.

Example II: A certain person may have a tendency to misspell certain words. He may then want the computer to help him with just the misspellings which are a problem for him.

In both cases, there is a need for the human to be able to modify the error checks in the computer according to the specific needs of a special work situation or a special human user of the computer. As before, the human using the computer needs power to influence the way the computer operates.

Certain error checks will detect obvious logical errors (like negative mass). Other will find things which probably are errors (like a wage decrease). The latter tests are often very useful, but they will depend on the circumstances more than the logical tests, since a piece of data which is highly unlikely one day may be much more plausible another day.

When an error occurs or is recognized, the human is aided if the consequences are not unnecessarily grave. A machine-malfunction should mean the loss of a few seconds of human work, rather than many minutes or hours of work.

For trivial errors, the consequences should be trivial, for serious errors, the consequences may have to be more troublesome. Thus, there should be several levels of errors. All errors should not immediately abort the program or the computer system.

When the execution of a program enters a stage where especially severe errors may occur, extensive error checking is advised before entering that stage. Example: If a human types in a command which would delete much information, then the computer might ask him again if he really is sure that this is what he intends to do.

Humans generally put rather high requirements on the reliability and repeatability of computers. They regard computers as technical systems, just like telephones or typewriters or railway trains or the distribution of electrical power. They expect the same kind of reliability from computers as from other technical systems. People also would like a computer system to give some kind of response even when the system is partially faulty. In the worst case, this response may just be a message about the error and the probable time of correction.

On the other hand, computer systems which helps a human in troublesome and problematic situations is often liked by people. When asking people why they like and are satisfied with good computer systems, they often say that the aid in avoiding and coping with errors is important. However, often the code to cope with errors and difficulties will be much larger than the code to handle normal, troublefree operation (See Duncansen 1971 and Hockenberry 1971).

For more information about how to respond softly to errors, see Randell 1975.

4.5 RESPONSIBILITY FOR ERRORS IN DATA

In computerized data bases it is often impossible to find out who is responsible for individual pieces of data. This means that humans who are hurt by erroneous data in computers find it difficult to defend themselves.

A solution to this would be to tag every individual piece of data with an indication of who is responsible for it. If such tagging is built into the data base handling system, it is not difficult to use. If an item of data is created by a computer program, then it should be tagged by both the person creating the input data and the person responsible for the program. Such tags might require more secondary storage memory, so this is a case where cost and efficiency requirements may come into conflict with reliability requirements. Sometimes, the information about who is responsible is stored in logs on cheaper media like magnetic tape. But this means that a question about who input what will be difficult and very costly to get an answer to.

A problem with such tagging is that it may increase the stress on the people inputting the data. Example: In a key-to-disk operation, the computer monitored every single mispunch made by the operators, and provided error statistics for the manager. This created a very difficult stress situation for the operators.

But this is an example of misuse of computer monitoring of who is responsible. The information about who input what should be there, but it should be used properly.

4.6 FITTING THE VARIATIONS OF REALITY INTO COMPUTERIZED MOULDS

A very common reason for humans to be frustrated with computers is that the computer program does not allow them to input or handle reasonably accurate descriptions of reality.

Everyone who has filled in a form knows that it is sometimes difficult to fit the things you want to say into the mold given by the form. There may not be a space for the thing to say, or every alternative answer may be misleading, or an answer may be misleading if it is not explained further, but there is no space for further explanations.

If the form was produced by a person or institution with certain values, then the form may mislead the answers according to the ideas of its producer.

Example I: A woman named "Alexandria" had too long a name to fit into the mold for "given name" in the system used by a hospital. Her name was shortened to "Alex", which meant that her sex was later mistakenly changed to "Male" which meant that her records disappeared. Example II: A file may contain information that a certain individual did at a certain time not pay a certain bill properly. But if there is no space to store the fact that the cause of this was a faulty delivery, then the information may give a very misleading impression of this person.

A nice partial solution to this problem would be if the computer allowed the user to affix an explanatory message to each individual piece of data in the data base. Other people might be allowed to add further explanatory tags. Important is that these tags do not disappear during the processing of the data. If certain data is created by computation from other data, then it would be best if any tags on the input data were also noted on the output data. But there is a problem when one output item is computed from many input items, like a mean value. The tags may be too many to list individually, but some kind of processing of them might at least be possible.

Time restrictions on the validity of data could also be added in similar ways, so that data items are not used at times when they are no longer valid.

5. MAN-MACHINE INTERACTIVE TECHNIQUES

5.1 COMPUTER GUIDED INTERACTION

Perhaps the most common kind of interaction is that which is guided by some form of questions from the computer.

Example:[1]
DO YOU WANT TO STOP NOW? no

The question from the computer can also be some kind of text describing what kind of input is expected.

Example:

```
INPUT THE TEMPERATURE,
FOLLOWED BY "F" FOR FAHRENHEIT OR "C" FOR CELSIUS: 23 c
```

A common kind of question is where the user is given a list of alternatives to choose between. Often he only has to answer with the number or the first letter of the chosen alternative:

Example:

```
DO YOU WANT TO
RETRIEVE INFORMATION FROM:
1    PERSONNEL FILE
2    ARTICLE FILE
3    NEWS FILE
4    POLICY DECISION FILE
     OR
5    UPDATE A FILE, OR
6    STOP.
```

This kind of selection between given alternatives is often called a "menu" conversation.

The advantage with computer guided interaction is that the language can be natural or almost natural human language, thus easy to understand for a human, and that the inputs from the human to the computer are still easy to interpret with a simple computer program.

The main disadvantage is that the human has rather limited freedom in how to perform his task, since he must follow the order of questions determined by the computer program.

In many cases, this restriction is a serious drawback, causing much dissatisfaction among humans using the program, but careful programming can give the user more freedom within the framework of computer guided interaction.

The methods of increasing the power of the user within the framework of computer guided interaction are: a) To carefully design the sequence of

[1] In the examples I will use upper case for what the computer writes, lower case for what the human answers.

questions so that the user has the choices he needs to do things in different ways at different times.

b) To include escape facilities, e.g. a special key to push to leave the normal sequence and go into a mode where commonly needed deviations from the normal sequence are available (e.g. to return to a previous question in the sequence).

One problem with the computer guided interaction is that the questions from the computer have to be rather verbose to be understandable by a user who has little experience with this program. But an experienced user may be thoroughly bored by the verbose explanations.

One solution to this is to have several modes of interaction, with different verbosity, which the user can choose between.

Another solution (often combined with the previous one) is the HELP facility described in the next section.

5.2 PUTTING THE MAMJAL INTO THE PROGRAM

Sometimes, when a user uses a program, he needs further explanations about what he can do with the computer at that point in the execution of the program. For this purpose, many programs are equipped with a so-called "HELP" facility.

The idea of the HELP facility is to have a special escape key or escape command, often "?", which allows a user at any time to get a more complete explanation of what to input. A HELP facility is very useful also in conjunction with other modes of interaction than the computer-guided.

The HELP facility can also choose between different explanations depending on the experience of the user. If the available HELP information is larger than what fits into the display screen, then it can be divided into screen-size chunks, and the user can be given a menu to choose what kind of help he wants.

The HELP facility should be designed together with the explanatory messages given when the computer does not understand what the user writes to it, and both are acceptable only when the program has been tested on many people, and the questions asked and comments given by these people have been collected in a file and processed by the system designers.

A carefully planned HELP facility, combined with good validity-testing and good explanatory messages for input not acceptable to the program, can mean that a program is usable with almost no prior instruction and that an inexperienced user always gets the help he needs, but still that the experienced user is not burdened with explanations which he does not need.

Since the computer will give the user the explanations he needs, the user does not have to consult a program manual as often. One can therefore regard this as a method of putting the manual into the program itself. (See Oskarsson 1975.)

Oskarsson says that it is important that an inexperienced user does not get into an input loop. Such a loop can happen if what he inputs is swallowed by the computer without explanations about what is happening, or when all input is rejected with insufficient explanations about why.

Example I: The computer waits for a special final character, and swallows all input until this final character comes. Even the help requiring "?" command may be swallowed. If the user does not know that the computer waits for this special final character, he cannot get out of the loop.

Example II: A user tries many possible commands, but they are all rejected by the computer without sufficient explanation. An ideal system should in such a situation automatically turn to simpler and more complete explanations.

5.3 FORM FILLING

With form filling, which is most common on display terminals, the user is presented with a "form" with blank fields on the screen of the terminal, and he is asked to fill in the fields.

Example:

```
SUBJECT   = ORDER
ITEM TYPE = pen
NUMBER    = 200
LAST DELIVERY  = 1975-06-16
CUSTOMER NAME  = johnson & co
CUSTOMER ADDRESS    = big street 5, smalltown
SPECIAL NOTES:
:                                             :
:                                             :
:                                             :
NEXT FORM WANTED : order
```

Form filling is really only a special case of computer guided interaction, but the fact that many questions are presented to the user at the same time gives him a better overview of the situation. He often also has certain freedom to move within a form.

Certain computer systems are designed in such a way that the computer processing of even the simplest input from the user is delayed several seconds. With such systems, form filling is often used. In such a case, the filling in of the form is handled entirely at the terminal, so that an interaction with the slow computer only occurs when a form is full.

However, this means that the power of the computer is only available to the user when the form is fully filled in. Error checking of the input data is for example not performed immediately, and special functions like escape and help functions are only available when the form is ready or in some systems by temporarily leaving the form. Better would be to design the computer to be able to respond immediately to simple inputs from the user. Users tend to be frustrated with computers which take a long time to do simple things, but they accept long delays for actions which they understand are complex to do. People are especially frustrated with computers whose response time varies in an unpredictable way.

Another disadvantage with form-filling without computer contact except at the end of a form is that other ways in which the computer can aid a user will be more difficult, like for example having the computer automatically fill in parts of the answers, depending on what was input previously.

A problem with form filling is that forms are often designed in such a way that the space for data is sometimes insufficient. Also, the user is often not provided with enough possibilities of inserting explanations and comments in cases where there is no straightforward answer to a certain subquestion in the form.

5.4 COMMAND DRIVEN INTERACTION

Another very common interaction mode is that in which the computer is driven by commands from the user.

Examples:

```
set speed to 300 m/s

display order summary

search newsfile for insurance and ( europe or america )

display number of sales employees for each department
```

A typical command begins with the name of the command. After that follows further information on how the command is to be performed. This further information can be binary choices, parameter values or formulas in some kind of formula notation. Often, the amount of possible information is very large. Since it is difficult for a human to formulate complex commands, default values are assumed for certain information if that information is not given by the user.

In natural human language, defaults play a very important part. We do not say "Every human being on board the ship was seasick on the first of August 1975", we just say "Everyone was sick", and this statement, taken in context, transfers the same message to the listener.

However, the disambiguation of defaults requires a mental process which is sometimes difficult to put into a computer.

If defaulted information is misunderstood by the computer, then errors may occur. Because of the limited facilities for disambiguation in most computer programs, they can accept less defaults than humans can in natural language conversations.

The defaults will often vary from user to user and from time to time. A system which allows the user to influence the process of defaulting may therefor be more satisfactory to the users than a static defaulting algorithm.

A good command interpreter should, if the user omits certain compulsory information, ask him for that in a series of questions after the command.

The same information to the computer could thus be given within the same system in one of the three following ways:

```
search newsfile for insurance after 1975-06-01

search newsfile after 1975-06-01
FOR WHAT? insurance

search
WHICH FILE? newsfile
FOR WHAT? insurance
ANY TIME RESTRICTION? after 1975-06-01
```

The command driven interaction can give the user more power, since he can choose between a number of commands and he can himself choose the order in which the commands are given to the computer.

However, the command driven interaction requires that the user is very familiar with those commands he needs to use, and is thus best for an experienced user.

With command driven interaction, it is much easier to provide the user facilities for going back or changing his mind. He simply uses the same command again or a special resetting command to change previously given data.

The program should be designed as much as possible so that the different commands can be given in random order by the user. One should as much as possible avoid letting the program go through several "stages" with different sets of commands at each stage, except where the user is free to move back and forward between the stages.

5.5 THE PROGRAMMABLE INTERFACE

A command driven interface gives a user a selection of commands to perform the tasks at the computer. But very often, a user wants more powerful commands because of his special needs.

Example I: A user very often needs to give the same series of commands in succession, and he does not want to have to repeat them all every time.

Example II: A user wants default values different from the usual ones for parameters to a certain system command. Example III: A user wants special error checks added to the interpretation of certain of his commands, e.g. checks that he did not omit certain information which normally is defaulted, or checks on the allowed values of certain command parameters.

Example IV: A user wants the computer to remember certain previously entered data, and use it automatically again for part of a new command.

Example V: A user wants an extra command to be automatically given to the computer every tenth time he gives another command.

As much as possible of such special user requirements should be able to put to the computer without any special programming.

If a user wants to add new types of data to the data base, then this should for example not require any explicit programming. The ideal system would just accept the new data and establish the necessary structures.

If a user wants to get data from the data base reported in a new way, then this should not either require any programming in the ordinary sense. A nice partial solution to this is used in the EMISARI system (See Renner 1973). A document in that system can contain within it special commands to include, at that point in the document, information taken from other documents.

Sometimes, however, programming is necessary to satisfy special user needs. Since this programming is very dependent on individual user requirements, and sometimes varies from day to day, the person doing the programming should be easily available to the user, sometimes maybe be the user himself.

It is n o t acceptable to input the user requirements to a central systems group which may take a long time before they can honor the request. The additional programming should be simple to make for someone who is not an expert on the inner workings of the system, and the additional programming should not endanger the safety of the system.

The solution to this may be to include in the command language of the computer system a special programming language for writing such simple additional user commands.

This language should have available certain simple programming tools plus all commands which the user is allowed to make directly.

Since programs in this language can do nothing except make ordinary user commands, they will not endanger the safety of the system in the way which ordinary programming in the system does.

The additional language should be recursive, so that programs written in it can use commands previously defined in the same language.

When the users themselves, or people close to them, have evolved very useful new commands in such a language, the systems people might add these commands to the central part of the system. In this way the user language can serve as a communication link between users and systems people.

Hans Karlgren has pointed out that is analogous to what happens in ordinary natural language. One person invents a new construct and begins to use it. If other people find it useful, the new construct may be adopted by more people, and some such new constructs become a permanent new part of the language.

Such a user language will very much increase the power of the users over the system. But it requires that one person close to each user group is trained in writing programs in the user language. Experience with computer systems have shown that so-called "local experts" play an important role. Local experts are local people, close to the users, who because of interest and ability have learned more than other users about the system. The programmable interface is a tool to help these local experts, so that they can better help themselves and other people who are using their abilities.

Because the user language increases the power of the users, it will better satisfy their human needs.

An inexperienced user can use the basic commands and user language commands written by someone else for him, and a more experienced user can learn to use the user language himself. Thus, a user language allows the users to expand their own skills and to use their new skills to adjust the environment to their own needs.

This kind of user languages are common for applications, where a user works with a set of data, on which he needs to perform various operations. He may want for fit his data to a certain shape, or experiment with difficult variations of the data etc. Examples:

1) A writer wants to test variations of a text.
2) A designer wants to test various designs.
3) A statistician wants to test various treatments of statistical data.

4) A budget maker wants to compare variants of his budget.
5) A planner wants to test different variants of his plan.

For this kind of applications data is kept in a workspace. Single commands, or series of commands in small routines, are executed by the user at the terminal. The commands operate on the data. This is very different from the activity of a programmer who wants to develop large program products. The routines are instead often written for once-only use, and a large task is divided into several small routines prepared one-at-a-time by the user. This way of working at a terminal has become very popular, because the user has close control all the way of what is happening. This is an important cause for the popularity of the well-known programming systems APL, LISP and TECO.

Programmable user interfaces are discussed further in Palme 1975.

5.6 NATURAL LANGUAGE INTERACTION

Some advantages with natural human languages are:

- No special training in a specialized command language is necessary.
- The range of what can be said in the language is very large.

However, there are also difficulties with natural human language:

- Compared with well designed artificial languages, natural language require longer texts to say the same thing. This is important if a user has to say almost the same thing hundreds of times a day to the computer.
- Natural language statements are often vague or ambiguous. This is often an advantage in talking with humans, but may be a disadvantage with computers because of the risk of misunderstandings.
- Programming a computer to understand natural language is a very difficult task.

Natural language is not at all always a superior language for man-computer interaction. In situations where the users have much training in the use of the computer, a more compact language can be better for them. However, specialized languages, difficult to use, will restrict access to the computers to those who have this special training, and to those who have access to assistants with the special training. People will be divided into classes, those who have access to the computer and those who have not. And this kind of class division can be dangerous.

The class division furthers centralization of power, since access to information is so central to power. People who are affected by the computer, but who are denied access to it, may often not be happy with it.

Sometimes one gets the impression that computer systems are intentionally written so that their language is difficult to interpret, just because the designers want to restrict understanding to a small group of people.

A well designed computer guided interaction can easily use natural or almost natural language, since the messages from the computer are in natural language, and since the user answers are short words which are easy to make the computer interpret.

A smooth, natural-language like user interface to a program can mislead the human into believing that the program is more intelligent than it is. Many people believe computers to be more clever than they are, and it is important to avoid enforcing this untrue belief.

Certain computer programs takes input in an artificial language which is carefully chosen so that programs written in it look like natural language. Examples of such languages are Cobol and Simscript II. The advantage is that it is easier for non-experts to read such texts. However, it is not necessarily easier for non-experts to produce the texts, since they may easily be mislead into believing that the computer actually accepts natural language, and not just a restricted artificial language which looks like natural language.

With command driven interaction, natural language becomes more difficult to use. No one has yet succeeded in writing a computer program which can understand natural language nearly as well as a human. It has been proved, that such a program must have not only linguistic capabilities but also knowledge about the world and reasoning capabilities. Only by using such capabilities can a computer correctly interpret even the simple word "they" in the following texts:

```
The soldiers shot at the women.  They fell down.

The soldiers shot at the women.  They used machine guns.

The city councilors stopped the demonstrants, because they
feared violence.

The city councilors stopped the demonstrants, because they
were going to burn an American flag.
```

Programs to understand natural human languages have thus been much more successful if they are only designed to accept phrases about a restricted subject area, like a time table, or a street plan of a city.

Here is an excerpt of a conversation with such a program:

```
where is the nearest restaurant?
```

```
THE DISTANCE TO BALDAKINEN AT DRAGARBRUNNSGATAN BETWEEN
S:T-PERSGATAN AND VAKSALAGATAN IS ONLY 500 METERS.

how can you go to the Galejan with a car?

FOLLOW KUNGSGATAN IN THE NORTH-WEST DIRECTION. TURN TO THE
LEFT AT S:T-OLOFSGATAN. etc.
```

The way such programs are usually written nowadays are that they recognize certain key words in the sentence, and that they have some kind of semantic data base to which they try to relate the concepts in the given sentence.

In a conversation between humans, each speaker adjusts his language to the others abilities, and questions are asked when the listener does not understand.

The same is to some extent possible in a conversational program, where the human user can adjust himself to the rather limited capabilities of the natural language understanding program, and where the program can ask questions when it does not understand.

However, there are not many computer programs which can adjust their language to the capabilities of the human except in rather simple cases. But even such rather simple adjustment can be a large improvement compared to no such ability at all.

A way of avoiding misunderstandings is to let the computer rephrase what it was told in other words. The human will then easily see if the computer has misunderstood something.

Natural language is therefore much easier to use in a conversational environment than in an application where the computer is expected to read pages of natural language text.

For further information about natural language understanding, see Palme 1973B which also contains a larger bibliography of relevant material.

Even if a system does not use full natural language, a human is greatly aided if the language is reasonably close to natural language. Certain computer system designers seem to love numerical codes for things easier said with normal words, even though a computer can as easily be programmed to read words as numbers.

However, there may be larger risks for misunderstandings with a computer understanding a limited natural language than with a computer understanding a more strict artificial language. One reason for this is that the human may be led to believe that the computer understands more than it does, another that the computer program may have to make uncertain guesses.

5.7 SAVE-RESTORE FACILITY

If a human has been sitting for a long time in front of a computer terminal, then that human will naturally not be happy if he has to re-do everything he has been doing during the terminal session.

However, re-doing everything may be necessary if he has made an error previously during the session, if the only way out of the error is to restart from the beginning and make the computer forget everything that has been input.

Two ways of solving this problem are the save-restore facility and the back-up conversation file.

With the save-restore facility, information about the status of the computer is saved at regular intervals, either automatically by the program or by commands from the user.

The user is then allowed to back-up the computer to the status at any of the previous check-points.

With the back-up conversation file, all conversation between the human and the computer is saved in a special back-up file.

The user can then, if he so wants, restart the program but with input from the back-up file instead of from the user directly. By using only part of the back-up file, or by editing the back-up file before restart, the user can correct the error without having to repeat the whole conversation.

Fully covering save-restore systems are very different or impossible to make, especially if several people have been using the computer system simultaneously and only one of them wants to back up. But even imperfect systems can be a great help. If the computer or program is not fully fool-proof, but has a tendency to go down, then the programs have to be designed in such a way that as little as possible is lost. If data are stored on secondary storage, and if primary storage is lost when the system goes down, then the programs can be designed in such a way that all information is moved into secondary storage and the secondary storage files are closed before every interaction with the user at the terminal.

Then, only one interaction will be lost at the system failure.

6. DATA BASE HANDLING SYSTEMS

Information in computers is more and more often stored through data base handling systems. This is intended to give more general-purpose systems, where more kinds of different information can be stored without difficult reprogramming every time you want to store new kinds of information in the data base.

However, many existing so-called "DATA BASE MANAGEMENT SYSTEMS" (DBMS) are strongly oriented towards data bases with rather fixed structures. For example, the well-known CODASYL DBMS proposal describes a system where the programs have to be changed and recompiled every time they want to access a new kind of data. Even the wish to look for information in the data base in a new way will often require program changes and recompilations. The advantage with systems like the CODASYL DBMS is however that old programs n o t using the new data do not have to be modified when more data is added to the data base. This was necessary with many older systems.

A common problem with data base systems is that they become overloaded with data, because each user group and each application has its own data needs. But other users och groups find it cumbersome with the extra data, and the centralized data base manager may not be able to allow every user group to store what it needs in the data base. The problem here is that a local user group or a special application cannot create its own special data base without either interference problems with other users, or problems to use the local data together with the central data base. The data base systems in the future must provide facilities for local data base, closely coupled to the centralized data base, but which do not interfere with groups or applications which do not need them. An undecided question is whether such local data bases should technically be separate data bases or part of the general data base.

There are two basic kinds of data base systems, relational data bases and text-oriented data bases.

Relational data bases look at data as logical relations. For example, the statement that "John is twenty years old, earns 800 dollars a month and is married to Anita" is regarded as a relation between "John", "twenty", "800 dollars" and "Anita".

Relational data bases allow the writing of programs which can directly answer factual questions like

"Is there any free seat on the 11.33 train to Stockholm?" or

"Has the order for three cars to Johnson been processed?".

The disadvantage with many relational data base systems is that they require data of very fixed structure and thus do not give very much freedom for saying things which do not fit into this structure.

Text-oriented data base systems regard the data basically as text. Sometimes, a record in the data base can contain several text items with different labels, like an "author" field, a number of "keyword" fields and a "message" field.

In the text-oriented system, a user can not always get a direct answer to a factual question. He can instead get an indication that some text items in the data base might provide him with the information he wants. There are also systems which try to combine certain ideas from both relational and text-oriented data base systems.

A very interesting example of such a system is the EMISARI system designed by Murray Turoff and described in Renner 1973 and Turoff 1975. There are other systems based on similar ideas.

These systems regard computerized data bases as primarily a medium for exchange of information between humans. Thus, the text in the data base consists of news items, policy decisions, factual information, messages etc. exchanged between humans.

However, the system also allows the handling of tables and other more formalized items.

A piece of text in the system can contain a command which means that parts of other text items are inserted into the text when it is typed out, so that e.g. the text "Weekly report" may print out different material every week without changing it.

In EMISARI, the users store messages which are of interest to others in the organization. They can also send messages personally to just one receiver.

When a user logs onto the system, he first gets any pending messages to himself (if he wants them) and is then allowed to store information or search for information. Search is done by author, by date limits, or by keywords. One can for example search for any news item on insurance within the last week.

Groups of people with interest in a common subject can have computerized discussions going on in the system.

An important property is that any user of the system can store and retrieve information. Every user will thus have immediate benefits from the system. This means that the users like the system much more than systems where they just have to feed in information but seldom get anything they need back from the computer. Another important property is that every item, even every single figure in a table, can have a tag with an explanatory message or a comment. Other people can add further messages, so that a long series of messages can in theory develop out of a single controversial item in a table.

Systems like EMISARI are especially useful in large companies or institutions, especially if they are spread over several geographic locations. The systems make people in the institution more aware of what is going on in other parts, and make it easier for them to get necessary information about what is happening.

The EMISARI system was used at the USA "Office of Emergency Preparedness" which had to cope with administrating a wage-price freeze across the USA. Administrative people without computer experience could very easily master the system and use it at short notice for a new crisis type problem. The system provided a means for interchange of information between employees all over USA. Bell Telephone in Canada has also used a similar system very successfully to exchange information between its employees spread all over Canada.

I personally believe that we will in the future get systems which within one system combines the facilities of both relational and text-oriented systems. A text can for example be used to explain a certain item in a relational data base.

Example: In Norway, a data base is going to be created of all dangerous chemicals used in industry. The employers only want scientifically proved facts to be stored in the data base, and that only the medical officer should be allowed to use the terminals. The employees want their representatives to be allowed to access the data base. They also want each local trade union representative to be allowed to store directly local experience with chemicals into the data base, together with his name and address, so that other employees at other companies can get his experience through the data base, and, if necessary, contact him by telephone or letter. This example shows how the employers prefer a strictly controlled, limited information transfer, while the employees prefer a freer and less limited communication.

7. LARGE COMPUTERS OR MINICOMPUTERS?

There is a growing trend of going over from large, centralized computers to local minicomputers. Apart from the technical reasons for this trend, an important reason is often that this is a way of distributing power over the computers. A local group can manage its own computer and the power of the centralized computer department decreases.

However, the advocates of larger computers argue that these systems can be arranged in such a way that each user or user group decides how they use the computer. They say that it is easier to program a large computers which has better programming tools, and that it is easier to implement more advanced programs, with better reliability and usefulness on larger computers.

The advocates of large computers say that an important goal of most computer applications is the communication of information between humans, and this communication is easier if all the humans are using the same large computers. They claim that users of mini-computers are isolationists. The mini-computer advocates counter that networks of small computers can do the same thing.

Of course minicomputers are best for some applications and large computers for other applications. What I want to point out with the above discussion is that minicomputers are not always the best for the distribution of power to local groups.

8. BIBLIOGRAPHY

Brännström, Jan 1975 and Mårtensson, Lena:	Gör enformigt jobb omväxlande. Arbetsmiljö 3/75 p 40-41.
Chapanis, Alphonse 1975:	Interactive human communication. Scientific American, March-April 1975.
Duncansen, L. A. et al 1971:	Interfaces with the Process Control Computer. In "Interfaces with the Process Control Computer", published by the Instrument Society of America.
Johnsson, Bengt & Andersson, Bertil 1974:	Man-datorkommunikation med teckenskärm, Projektrapport nr 1. LIH-ISY-R-0001, Linköpings Högskola, Inst. för systemteknik, S-581 83 Linköping, Sweden.
Gordon, Thomas 1970:	P. E. T. - Parent Effectiveness Training.
Gordon, Thomas 1974:	T. E. T. - Teacher Effectiveness Training.
Grip, Arne 1974:	ADB-system och kommunikation. Hermods-Studentlitteratur, S-221 01 Lund 1, Sweden, 1974.
Harris, Thomas A. 1969:	I'm OK - You're OK. (Swedish translation published 1974.)
Hoare, C. A. R., 1975:	Software design: a Parable. In Software World, vol. 5, Numbers 9 & 10.
Hockenberry, Jack et al 1971:	The Human Environment Misfit Factor Concept. In "Interfaces with the Process Control Computer", published by the Instrument Society of America. Kent, Jan 1975:
Kent, Jan 1975:	Datamaskinassistert undervisning for ikke snakkende barn. Presentation at NordDATA-75.
Leijonhufvud, Sten 1973:	Introduktion till strukturering av dialoginriktade datorprogram. FOA P rapport C 8361-M3, Mars 1973.
Marguiles, Fred 1976:	Not for the people, but with the people. Data no. 10, 1976 pp 17-19.
Mumford, E. and Sackman, H. (editors) 1975:	Human choice and Computers. North-Holland publishers, Netherlands, 1975.
Martin, J. 1973:	Design of man-machine dialogues. Prentice-Hall, Engelwood Cliffs, N.J., 1973.
Nygaard, Kristen 1974 and Bergo, Olav Terje:	Planlegging, styring og databehandling. Tiden Norsk Forlag 1974.

Ohlin, Mats 1975:	SAFEIO - system for safe and fast conversational SIMULA programming. FOA 142, July 1975.
Oskarsson, Östen 1975:	About design of conversation interfaces for non-expert users. Datalogilaboratoriet, Sturegatan 1, S-752 23 Uppsala, Sweden, January 1975.
Palme, Jacob 1973:	The SQAP data base for natural language information. FOA P Report C8376.
Palme, Jacob 1974:	The General Public Democratic Information System. Data, Copenhagen no. 3, 1974 pp 22-24.
Palme, Jacob 1975:	REAL TIME - easing the user task. In Software World, Vol. 5, No. 8, pp 9-11. (Also published as FOA HE 1 report C10001, July 1974, under the title "A method of increasing user influence on computer systems".) Palme, Jacob 1976:
Palme, Jacob 1976:	Internationell kurs om interaktion mellan människa och dator, FOA 1 Rapport C 10056-M3(ES,H9), Oktober 1976.
Palme, Jacob 1977:	A Man Computer Interface Encouraging User Growth, FOA 1 Report C 10073, September 1977.
Randell, B. 1975:	System Structure for Software Fault Tolerance. In proceedings of the 1975 international conference on reliable software, pp 437-449.
Renner, Rod L 1973:	EMISARI - A Management Information System Designed to Aid and Involve People. In Proceedings at the Fourth International Symposium on Computers and Information Science (COINS-72), Miami Beach, Florida, December 1972, also available from the Office of Emergency Preparedness, White House, Washington D.C.
Shackel, B. 1969:	Man-computer interaction - the contribution of the human sciences. Ergonomics, 12:4 (July 1969) pp. 485-499.
Seligman, Martin E. P. 1975:	Helplessness: On depression, development and death, by Martin E. P. Seligman, W.H. Freeman, San Francisco1975.
Turoff, Murray 1972:	"Party-line" and "discussion" computerized Conference Systems. In proceedings of the International Conference on Computer Communication (ICCC) 1972.
Turoff, Murray 1975:	The Future of Computer Conferencing. The Futurist, Vol. IX, NO. 4, August 1975, pp 182-195.