which makes it impossible for you to drive when your blood alcohol level is above a certain level. And suppose there is an exceptional case. John has a heart attack, and the only person available to drive to the hospital, Mary, has drunk a little too much. The computer stops Mary from driving the car, and John dies. They are in remote cottage without telephone and outside the range of cellular phone networks.

My argument: This is an example of a computer making a decision. The computer decides that Mary is not allowed to drive the car, and in this particular case, the decision made by the computer might be ethically wrong. This is no easy issue, the best may still be to have such a computer device in the car. But the example illustrates the problem and the danger of programming computers to make decisions about right and wrong.

Counter-argument: No, the computer did not make any decision. The decision was made by the humans who programmed the computer. They may have weighed pros and cons, and decided that the advantage with such a drunk-driving-protection device is worth the risk that in some exceptional cases the outcome may be wrong.

My counter-counter-argument: This is becoming a discussion of the meaning of words. You do not accept that the computer made a *decision*. OK, let us then say that the computer made a *ruling*, or whatever word you prefer to apply to the case where a computer prohibits you from doing something. You are avoiding the ethical issues: In what way should we program computers to control human beings.

I am not a fanatical liberal who is against all laws and rules. I am quite willing to accept that in some cases it may be ethically right to program computers to prohibit you from drunken driving or stop children from downloading bomb-making recipes from the Internet. But I am advocating that in those cases you are programming the computer into making *decisions*, or *rulings* or whatever word you prefer to use. And this can be dangerous and you should be aware of the risks.

Solution: In this special case, a solution might be to allow the driver of the car to communicate with SOS Alarm, and allow them to send a code to override the breathanalyzer lock in some very exceptional cases. By doing this, we are moving the final decision from computers to humans.

Explanation of Problem

One mode of human communication is the setting of rules. Some human beings make a list of rules. The rules may be a law, a local ordinance, ethical rules of a professional organizations, company rules for employees or published in other ways. The human beings may also introduce ways of enforcing the rules, such as courts of law, committees on ethical conduct, etc. This is sometimes (not always) necessary even though the rulings made are sometimes wrong, like convicting innocent people. But we accept that this risk must be taken because without law and order society would not work.

An example: Even if it is forbidden for a pedestrian to step into the street against a red light, there are special cases where this rule does not apply. Suppose a child runs out on the street, and the only way to stop the child from getting run over by a car is to run out and catch the child. Such special cases are easily handled by humans. No human court would sentence a person for running towards a red light in such a case. But it is not so easy to teach a computer to understand such exceptions.

The danger is that people do not always understand, that putting such rules or laws into a computer, and programming the computer to enforce the rules, is something very different from having humans implement the rules by human decisions. Humans can understand the special conditions of special circumstances. A human might decide, in the example above, that in this special case the importance of getting John to a hospital is higher than the risk of Mary driving while intoxicated.

Counter-argument: A counter-argument made at [Martin 1997] was that in this case it was humans who made the decisions, by programming the car computer, but their decisions were wrong. They did not take all circumstances into account. They should have made a more advanced program, which could take into account the special circumstance of the heart attack situation.

Counter-counter-argument: What you are doing, with this kind of argument, is to make the computer program more advanced and complicated, to reduce the risk that the decisions/rulings made by the computer are wrong. The path you are treading may make things worse instead of better. More complex and advanced computer rulings may increase the risk of wrong decisions instead of reducing them. The right solution may sometimes instead be to accept that the computer is not perfect, and thus that all rules do not have to be enforced by computers alone.

"Italian Strike" Example

A well-known method for strikes is to continue to work, but to adhere 100 % to all rules while working. This rapidly causes many businesses to a complete standstill, or at least makes them work much slower and less efficiently before. The reason for this is that even good and benevolent rules can have disastrous effects if adhered to 100 %, the way a computer would do if programmed to enforce them.

Filling in a Form Example

If you fill in a form manually, you have the option of adding an accompanying sheet of paper explaining why you filled in the form in a particular way. In most forms, where there is a ”yes-no” question, it is quite possible to omit the checkboxes and write a more nuanced explanation below or above. The human who receives the form will read and understand and interpret this. When the form becomes computerized, this freedom to not follow the prescribed way of filling in the form often disappears.

Society Evolves by Many People doing Things in Better Ways

An important way in which human society is evolving is that many people make small and large decisions to try out new and better ways of doing things. If the computer program stops them from doing things in other ways that those specifically allowed, this will prevent people from finding better ways of doing things, and thus stop imrovements.

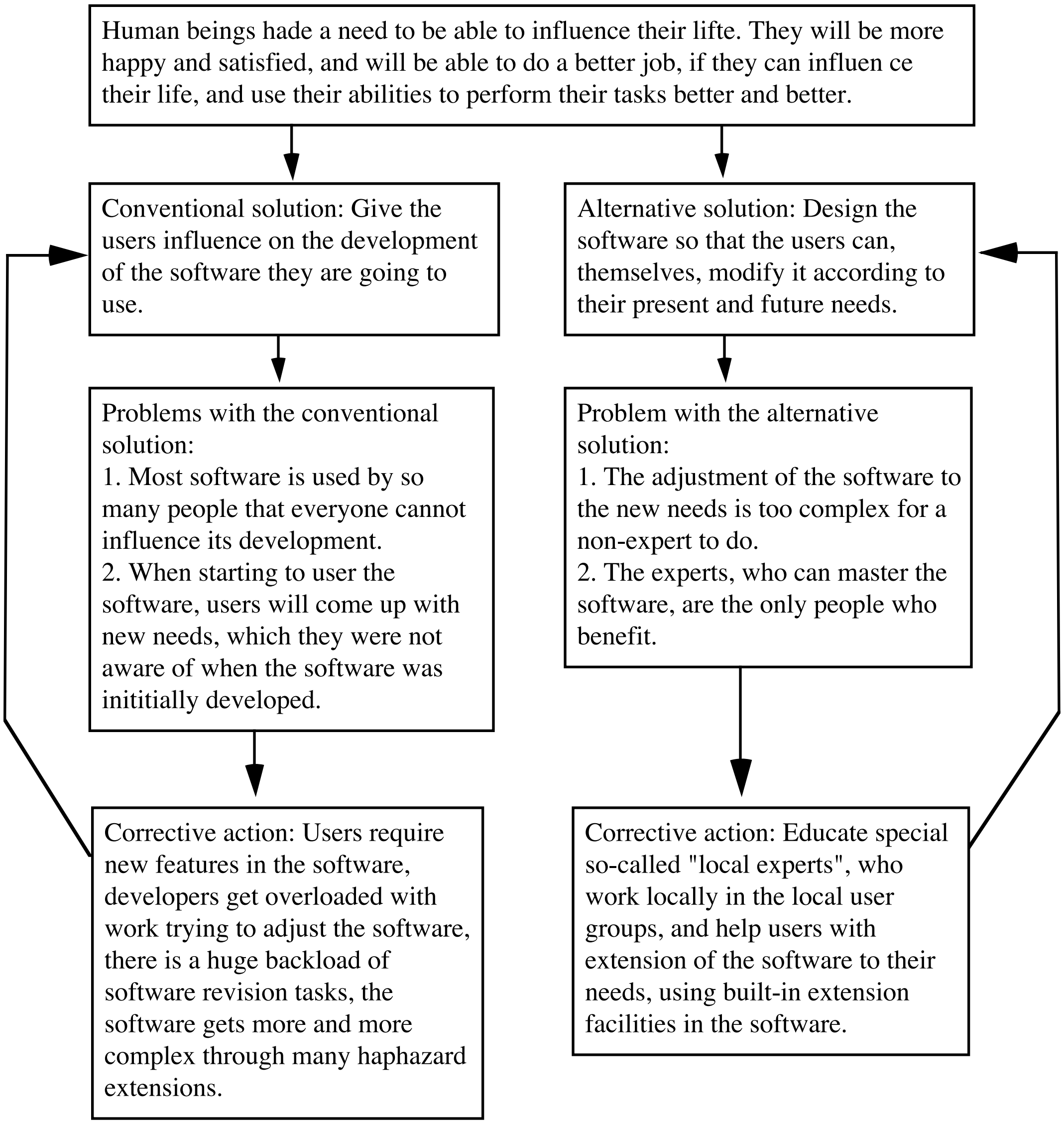
Sometimes this may be necessary. For example, there is a human tendency to stop performing actions which are necessary only to avoid seldom occurring risks. Example: A pilot forgets an item on the pre-flight check list, or a night watchman forgets to go to a normally empty part of the building. In such cases, it may be necessary to use technical means to ensure that the human follows the rules, for example the night watchman must turn a key to show that he has passed that part of the building. But this does not forbid the night watchman from disobeying the rules in special cases, for example skip the empty corridor if there is a thief in another part of the building. The danger is when the computer does not allow you to do things in other ways than those foreseen when programming it.

Two-sided Communication is Better than Enforcing one Solution

It is a well-known fact that power is addictive. That is why we design human societies with so many safe-guards against giving individual people too much power. We must understand that the power to control other humans by design of computer software can also be addictive [Hoare 1975]. Only by understanding this, can we stop people from putting too much control of humans into their software. I am not arguing that there should be no control of humans by software. The common "Are you sure?" dialog boxes are often motivated, even though they are sometimes a nuisance. But those who design computer software should be aware of the risks of putting too much control of humans into the software.

This will reduce the possibillities for people to influence their environment, and will create a feeling of helplessness, which may cause dissatisfaction and depression [Seligman 1975]. When two humans communicate regarding a task, the outcome of their discussion is usually twosided or so called win-win solution, a solution where the needs of both are taken into account [Harris 1969, Gordon 1970]. When the task is controlled by a computer (even though a human did originally program the computer) the interaction necessary to achieve a win-win solution is often not possible. It iw well-known that such situations easily cause frustration and dissatisfaction and also often mean that a less good way of performing the task has to be used.

User influence



Human beings hade a need to be able to influence their lifte. They will be more happy and satisfied, and will be able to do a better job, if they can influen ce their life, and use their abilities to perform their tasks better and better.

Conventional solution: Give the users influence on the development of the software they are going to use.

Alternative solution: Design the software so that the users can, themselves, modify it according to their present and future needs.

Problems with the conventional solution:
1. Most software is used by so many people that everyone cannot influence its development.
2. When starting to user the software, users will come up with new needs, which they were not aware of when the software was inititially developed.

Problem with the alternative solution:
1. The adjustment of the software to the new needs is too complex for a non-expert to do.
2. The experts, who can master the software, are the only people who benefit.

Corrective action: Users require new features in the software, developers get overloaded with work trying to adjust the software, there is a huge backload of software revision tasks, the software gets more and more complex through many haphazard extensions.

Corrective action: Educate special so-called "local experts", who work locally in the local user groups, and help users with extension of the software to their needs, using built-in extension facilities in the software.

Stopping the Porn

An example of an application area, which illustrates the problems with computers deciding what is right and wrong, is the area of the porn-blockers, program modules meant to prevent use of the Internet for unsuitable purposes. They are used by parents who do not want their children to download porn on the net, by schools and libraries, and also by employers to stop employees misusing thir office computers, and even by countries to control the flow of information and stop undesirable information. The People's republic of China and Singapore are examples of countries who want to stop unaccceptable information, such as views by so-called dissidents.

These programs, however, have severe problems. Either they permit only access to listed and allowed sites. But since the developers cannot keep up with all pages on the Internet, only a small subset is listed. Alternatively, they try to guess whether a document is suitable or not, this is done by scanning for certain character strings, like “sex”, “breast” and “xxx”. This has led to horrendous mistakes, such as prohibiting information from Middlesex (a local government in England) or prohibiting information about breast cancer. A computer user complained that when he downloaded code in a particular script language from the Internet, the code was distorted in funny ways. For example, the following piece of script code:

```
#define one 1 /* foo menu */
#define two 2 /* bar baz */
```

Was corrupted in the following way:

```
#define one 1 /* foo me   */
#  fine two 2 /* bar baz */
```

I leave it to the reader to compare the scripts and conclude what “Cybersitter” had done with his script code and why.

Is the Internet Illegal

Actually, almost all usage of the Internet is illegal according to the privacy protection laws in many countries. These laws prohibit all transport of personal data from one country to another without permission from the government. If these laws had been programmed into the computers, then we could not have had the Internet we have today. We should be happy that the laws are enforced by humans who understand that the intention of privacy protection laws are not to prohibit free speech. You cannot be sure of this. I ran one of the first Swedish BBS-es in 1978. We were forbidden to run our BBS by the Swedish Data Inspection Directorate. Later on, we were allowed to start it again, provided that we did not allow anyone to write any political or religious opinions in forums on the BBS (Since the Swedish Data Act forbade the creation of registries of political and religious opinions, except in certain special cases, but in contradiction to the Swedish constitution, which specifically says that the right to communicate freely on political and religiuous issues should be protected). We continued to use our BBS including some discussions of political or religious issues. No one prosecuted us. But what would have happened if the computer had been programmed to recognize and automatically prohibit any message with political or religious content?

The Computer need not Stop all Unwanted Behavior

The idea that human rulebooks should be programmed into computers is closely connected to another faulty idea. This other faulty idea is that anything is legal, which the computer permits you to do. “The computer did not stop me from accessing this data”, is the standard defense from the cracker who breaks into a computer. If you believe that anything allowed by the computer is legal, then obviously you have to program the computer to prohibit all unpermitted behavior. One can understand the danger of this by trying to envision a society where all illegal acts are made impossible to perform. Hammers are not allowed, since you can kill people with a hammer. Suppose you need a hammer in woodworking. Tough luck, this is illegal, hammers are inherently dangerous. In order to prevent crime, every movement from one place to another without permission might be prohibited and monitored. Is this the kind of society we want?

A real example which I have actually seen: A building where I worked was split into zones. Whenever you moved from one zone to another, you had to insert a keycard into a slot to open the door. A person inserted the keycard, opened the door, then dropped the keycard, bent to pick it up, while the door closed with the person still in front of the door. The door locked automatically, so the person inserted the keycard again to open the door. This did not work. The computer obviously reasoned as follows: “This person has already passed into the new zone. She cannot be in front of the door. So her keycard must be falsified or wrongly used.”

Faulty programming of the computer? Perhaps, but you can never be sure that your program is perfect. And making the program more complex by taking into account more special cases in deciding what the computer allows and prohibits may introduce more bugs, while removing old bugs in the software. The new bugs may be more insidous and difficult to find. All problems are not best solved by making computer software more complex. Some problems are better solved by letting humans, instead of computers, make decisions!

Calendar Scheduling

A good example to discuss these issues is the use of computers to schedule meetings. This may at first seem like a good way of using computers. But the more you look at the problem, the more you find that real meeting scheduling includes so many special cases, where human judgment is needed, that it becomes very difficult to get the computer to do this automatically. For example, some meetings are more important than other meetings, and may cause other meetings to be rescheduled, but such a decision cannot be done by a computer. And there are contextual factors, like knowing that you should never schedule a meeting with a certain person on a Monday morning, which everyone knows, but which are difficult to put into the computer.

Conclusions

⇒ The successes of human society is based on the flexibility of humans and their willingness to adapt their activities to different circumstances.

⇒ Humans are most happy and productive if they can influence their living environment and contribute to solving problems together.

⇒ Laws and regulation are a form of communication between humans. They are in reality only guidelines, people have to adapt to varying circumstances and interpret and apply the rules with understanding and human compassion. If everyone had to adhere 100 % to all laws and regulations, human societies would not work any more.

⇒ This is usually no problem when the laws and regulations are written on paper. But if the laws and regulations are programmed into computers, so that the computers control what is allowed and not allowed, serious problems will often occur. In the best case, people will only be unhappy and unproductive, in the worst case, major catastrophs can occur.

⇒ Computer software must be designed to allow flexibility and human choice. Laws and regulations should be interpreted by humans, not by machines.

⇒ Making the software more complex, to include in it more different special handling of special circumstances, will often only make it worse. Instead of complex software, software should be flexible and open-ended.

⇒ There is a human tendency when designing software to want to include in it “proper procedure” and “experience how things should be done”. This tendency can easily produce unusable or unsuitable software.

⇒ Possible exception: Certain security rules, where enforcement is needed to overcome human weaknesses.

References

Gordon 1970: P. E. T. - Parent Effectiveness Training, by Thomas Gordon, 1970.

Grip 1974: ADB-system och kommunikation *(Data processing and communication)*. Hermods-studentlitteratur, Lund, Sweden, 1974.

Harris 1969: I'm OK - You're OK, by Thomas A. Harris, 1969.

Hoare 1975: Software Design: a Parable. In Software World, vol. 5, No. 9&10, 1975.

Martin 1997: Empowering Educators and Parents: Content Advisories for the Internet. By C. Dianne Martin, Proceeedings of the ITiCSE ACM conference, June 1997. URL: http://www.dsv.su.se/~jpalme/reports/iticse-notes.html#martin

Palme 1975 Interactive Software for Humans. At URL: (HTML version): http://info.dsv.su.se/~jpalme/reports/interactive-software.html and URL (Acrobat version): http://info.dsv.su.se/~jpalme/reports/interactive-software.pdf .

Palme 1975: Interactive Software for Humans, by Jacob Palme. An abbreviated version was published in In Management Informatics vol. 7(1976) pp 4-16. URL: http://info.dsv.su.se/~jpalme/reports/interactive-software.html

Seligman 1975: Helplessness: On depression, development and death, by Martin E. P. Seligman, W.H. Freeman, San Francisco1975.