

Commentary: Human error and the design of computer systems.

Communications of the ACM, 1990, 33, 4-7.

Donald A. Norman

In 1988, the Soviet Union's Phobos 1 satellite was lost on its way to Mars. Why? According to Science magazine, "not long after the launch, a ground controller omitted a single letter in a series of digital commands sent to the spacecraft. And by malignant bad luck, that omission caused the code to be mistranslated in such a way as to trigger the test sequence" (the test sequence was stored in ROM, but was intended to be used only during checkout of the spacecraft while on the ground) [7]. Phobos went into a tumble from which it never recovered.

What a strange report. "Malignant bad luck"? Why bad luck: why not bad design? Wasn't the problem the design of the command language that allowed such a simple deviant event to have such serious consequences.

The effects of electrical noise on signal detectability, identification, and reliability are well known. Designers are expected to use error-detecting and correcting codes. Suppose interference from known sources of electromagnetic noise had corrupted the signal to Phobos. We would not blame the ground controllers: we would say that the system designers did not follow standard engineering practice, and we would reconsider the design of the system so as to protect against this problem in the future.

People err. That is a fact of life. People are not precision machinery designed for accuracy. In fact, we humans are a different kind of device entirely. Creativity, adaptability, and flexibility are our strengths. Continual alertness and precision in action or memory are our weaknesses. We are amazingly error tolerant, even when physically damaged. We are extremely flexible, robust, and creative, superb at finding explanations and meanings from partial and noisy evidence. The same properties that lead to such robustness and creativity also produce errors. The natural tendency to interpret partial information -- although often our prime virtue -- can cause operators to misinterpret system behavior in such a plausible way that the misinterpretation can be difficult to discover.

Quite a lot is known about human performance and the way it applies to system interaction [1]. Several classes of human error have been identified and studied and conditions that increase the likelihood of error can be specified in advance [3, 4, 5]. Communication systems can be designed to be error-tolerant and error-detecting or correcting. In a similar way, we could devise a science of error-tolerant, detecting or minimization interactions with human operators [2].

Many advances have been made in our understanding of the hardware and software of information processing systems, but one major gap remains: the inclusion of the human operator into the system analysis. The behavior of an information processing system is not a product of the design specifications: it is a product of the interaction between the human and the system. The designer must consider the properties of all the system components -- including the humans -- as well as their interactions. The various technical publications of the field attest to a concern with

software and hardware, but emphasis on human functionality and capability is lacking. Many failures of information systems are attributed to human error rather than to the design. We are going to suffer continued failures until we learn to change our approach.

One of the first things needed is a change in attitude. The behavior we call human error is just as predictable as system noise, perhaps more so: therefore, instead of blaming the human who happens to be involved, it would be better to try to identify the system characteristics that led to the incident and then to modify the design, either to eliminate the situation or at least to minimize the impact for future events. One major step would be to remove the term "human error" from our vocabulary and to re-evaluate the need to blame individuals. A second major step would be to develop design specifications that consider the functionality of the human with the same degree of care that has been given to the rest of the system.

In the case of the Soviet Mars probe, the American journal Science wrote its report as if the incompetence of the human controller had caused the problem. Science interviewed Roald Kremnev, director of the Soviet Union's spacecraft manufacturing plant. Here is how Science reported the discussion: "what happened to the controller who made the error? Well, Kremnev told Science with a dour expression, he did not go to jail or to Siberia. In fact, it was he who eventually tracked down the error in the code. Nonetheless, said Kremnev, 'he was not able to participate in the later operation of Phobos' " [7]. Both the reporter's question and the answer presuppose the notion of blame. Even though the operator tracked down the error, he was still punished (but at least not exiled). But what about the designers of the language and software or the methods they use? Not mentioned. The problem with this attitude is that it prevents us from learning from the incident, and allows the error-prone situation to remain.

Stories of related failures of computer systems due to "human error" are easy to find in every industry: nuclear power, aviation, business, the stock market, and of course, the computer industry itself. In the August, 1989 issue of the Communications of the ACM, the following item appeared in the section News Track: "A computer operator at Exxon's Houston headquarters has been fired for inadvertently destroying computer copies of thousands of documents with potentially important information relating to the Alaskan oil spill. The ex-employee, however, says he is being used as a scapegoat and that none of the tapes he erased were labelled 'Do Not Destroy.' " The information provided about this incident is too sparse to form a conclusion, but if the system had been designed with the characteristics of human operators in mind, the preservation of tapes would not depend upon the existence of a simple (human-generated?) label "do not destroy." Thus, either the incident would not have happened, or the excuse would not have been plausible.

Perhaps it is time for the ACM to take the lead in this matter for the design of computational systems. There is considerable expertise among its members, including the Committee on Computers and Public Policy and one special interest group devoted to related issues (SIGCHI, the Special Interest Group on Computer-Human Interaction).

There is also a convenient place to start. On the electronic computer networks, Peter Neumann moderates the valuable Forum on risks to the public in computers and related systems, labelled as an activity of the ACM Committee on Computers and Public Policy. This RISKS forum collects, reports, and comments on incidents that include human error and design, but

these are not sufficiently precise or authoritative to be used for professional advancement of the field. The sources of the information are often reports in the media, reports that are incomplete, usually written before all relevant information is gathered, and subject to other sources of inaccuracies and biases. (The items from Science and the CACM's News Track that I cited exhibit all these sources of unreliability.) There is a lot of potential benefit to be gained through the careful study of design failures: other disciplines have learned to benefit through such careful review and analysis [6]. In reviewing the cases presented in the RISKS forum, why not use them as guides to better design?

There are several existing systems used in other industries that could provide a model. One major source of valuable advice in the aviation community is a collection of incidents known as ASRS, the Aviation Safety Reporting System, run by NASA-Ames, with a computer-readable database administered by Battelle. Here, people in the aviation community who witness or commit errors or other related problems write a description of the incident and their interpretation and mail them to ASRS. The ASRS investigators may call back to check on accuracy or get more information, but once the information has been confirmed and clarified, the part of the form that contains the submitter's identification is returned to that individual. ASRS also removes all identifying information to make it impossible for the particular submitter or incident to be determined. This anonymity is critical to the accuracy and completeness of the database. Because NASA has no regulatory power and has a good record for keeping the sources confidential, this database has become trusted by the aviation community. People are now willing to describe their own actions if they believe the report will be useful for the improvement of aviation safety. Many improvements in cockpit and other parts of aircraft design have been made by designers who have studied the patterns of errors that can be seen in this database.

A critical aspect of the ASRS system is that the reports are not seen by any supervisors of the submitters. Similar attempts in other industries have failed because their reports were submitted through a chain of authority that included the person's supervisor or plant management -- people who have biases to sanitize the report or to form negative judgements of the reporter. Thus, the incident reporting system for the nuclear industry is not an impartial guide to actual operating practices. Anonymity and self-report have worked well, along with a system of verification and clarification such as is performed by the NASA ASRS team (mostly composed of retired aviation professionals).

In similar fashion, the United States National Transportation Safety Board (NTSB) performs a detailed analysis of transportation accidents (aviation, highway, marine, railroad, and pipeline). These reports are extremely valuable and are a major force in the improvement of safety in the relevant industries. (The NTSB reports are, by statute, not allowed to be used in legal proceedings to determine culpability for an event. This kind of protection is essential in today's litigious society to allow the investigation to proceed without fear that the results will be misinterpreted or misused.)

Should the ACM sponsor similar initiatives? I don't know, for its issues are different from those faced by other industries. But I propose that the ACM investigate the possible ways of improving this part of the profession. The ACM could take the lead in establishing some positive, constructive

actions to elevate the human side of computing to a level of concern and respectability equal to that of the physical and symbolic side.

REFERENCES

1. Helander, M. Handbook of human-computer interaction. 1988.
2. Leveson, N. G. Software Safety: Why, What, and How. ACM Computing Surveys, 18, 2 (1986),
3. Norman, D. A. Design rules based on analyses of human error. CACM, 4 (1983), 254-258.
4. Norman, D. A. The psychology of everyday things. Basic Books, New York, 1988.
5. Perrow, C. Normal accidents. Basic Books, New York, 1984.
6. Petroski, H. To engineer is human: The role of failure in successful design. St. Martin's Press, New York, 1985.
7. Waldrop, M. M. Phobos at Mars: A dramatic view -- and then failure. Science, 245 (1989), 1044-5.

Don Norman <http://www.jnd.org>