

The subject of software production and quality was covered in a broad sense at Garmisch. The purpose of this meeting was to concentrate in particular on the technical problems of software; having got a specification, the next problem was to produce a program that satisfied the specification and was of good quality. It was divided into two main areas; a project consisting of the problem of producing the specified task (3.1) and its evaluation (3.2). The magnitude of the problem is indicated by the following quotations.

Hopkins: We face a fantastic problem in big systems. For instance, in OS/360 we have about 1000 errors each release and this number seems reasonably constant. Now why do we have these errors? Does this number relate in some way to the size of the system? The rate of errors is related to the rate connected with the system update/release cycle; initially people don't report errors because of too many problems. Engineers emphasize well-documented and rigorous methods and practices, develop and adhere to codes of ethics, and invest in tools that support their work.

Schorr: It is not quite right that the rate is constant; it seems in fact to be slowly increasing!

3.1 Correctness

Pivotal events: Revolutions or evolution?

Key moments in the history of the engineering culture often hide more complex historical developments.



NATO Software Engineering Conference (1968)

The term “software engineering” is born and everyone agrees on the problems faced by the growing industry, but there is little agreement on the right path forward.

3.1.2.2 On on-line and off-line debugging techniques

I would like to discuss the trend towards conversationality in our tools. There has been, since the advance of timesharing and on-line consoles, a very hectic trend towards development of systems which allow the interactive development of programs. Now this is certainly nice in a way, but it has its dangers, and I am particularly wary of the danger of the loss of the concept of the program as a whole. People who are used to the response of a computer might find it difficult to follow the progress of a program which is being developed in a conversational manner. I am particularly wary of the danger of the loss of the concept of the program as a whole. People who are used to the response of a computer might find it difficult to follow the progress of a program which is being developed in a conversational manner.

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

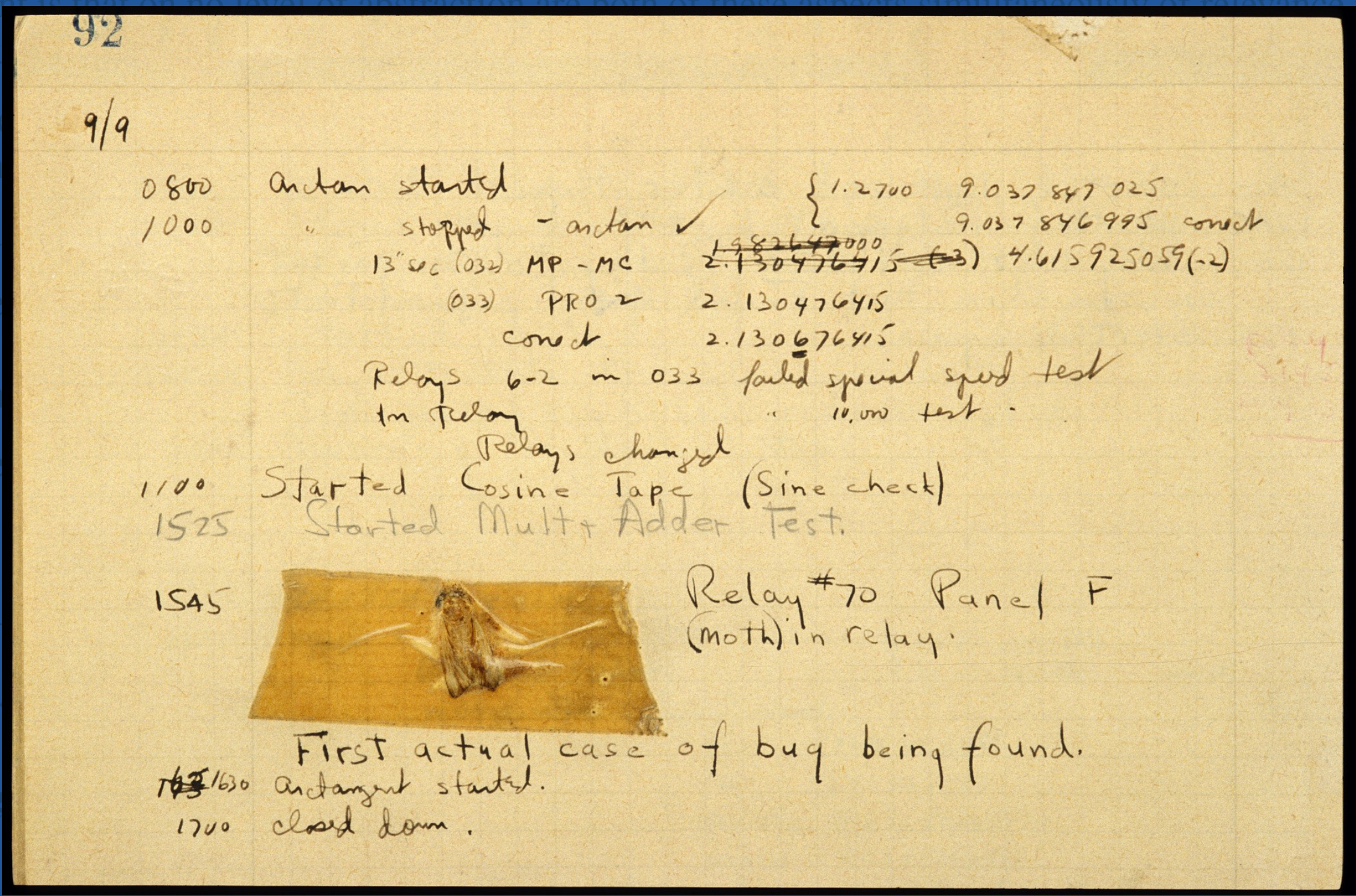
Individuals and interactions over processes and tools  
Working software over comprehensive documentation  
Customer collaboration over contract negotiation  
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

The Agile Manifesto, Snowbird meeting (2001)

Captured but also simplified engineering values that were developed over the preceding decade, drawing from ideas such as Scandinavian participatory design.

Choosing terminology: Bug or failure?



First actual case of a bug being found (1947)

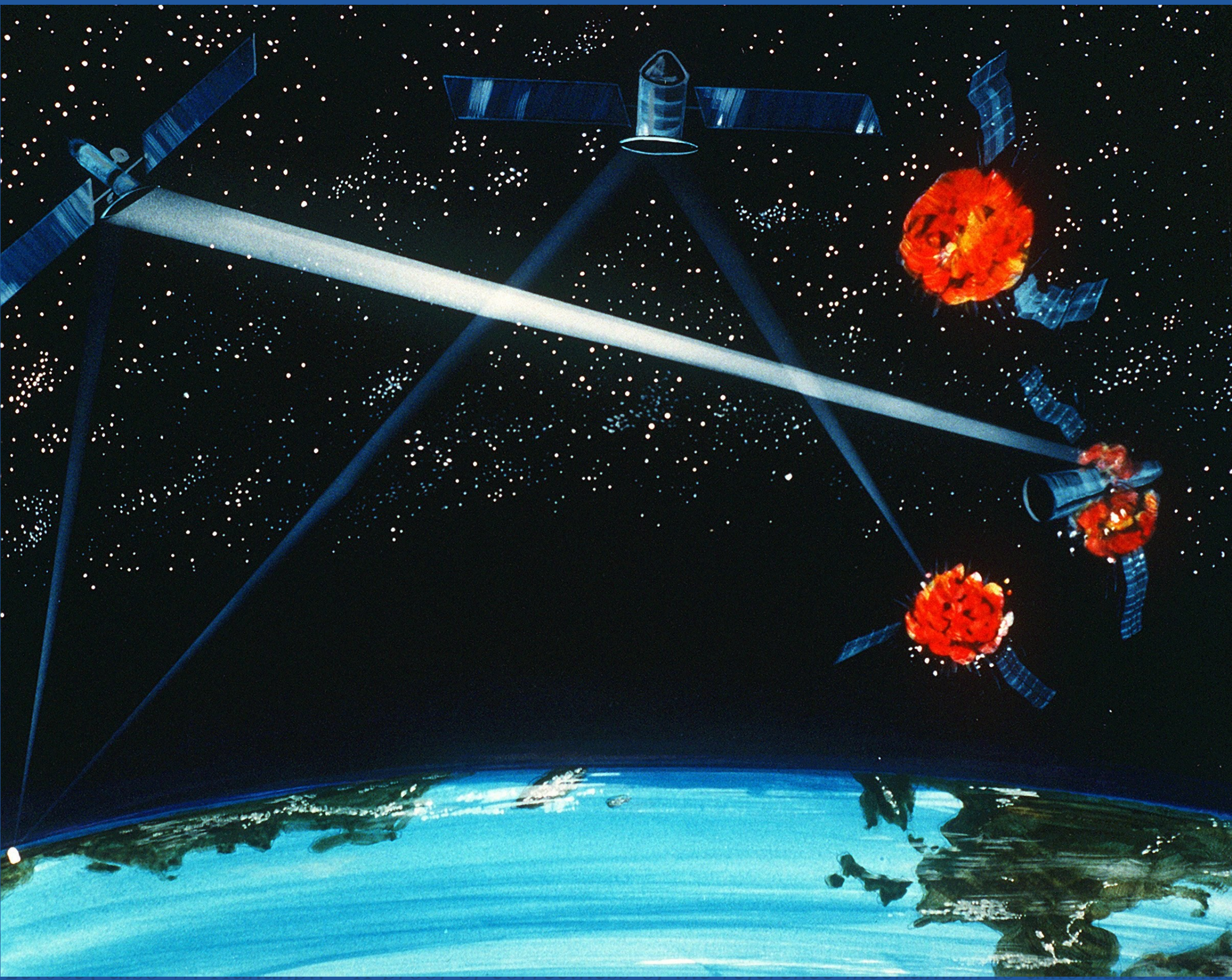
By using the term bug rather than failure, engineers suggest that the errors are small and easy to correct. This is in stark contrast with the \$300bn spent on fixing the “Y2K bug”

The paper by Needham given in section 7.10 is also relevant to the subject. Comment which, in the editors' opinion, relates specifically to that area defined by Aron as “testing” will be found in section 5.

4.2.1 On classical techniques

The nature and limitations of software

The development of the U.S. Safeguard antiballistic missile system required a new engineering language to answer the question: Can such a software system be reliably built?



Joseph Weizenbaum (1969)

Computer systems can only become reliable if “the environment that they are to control or with which they are intended to cooperate has a change rate smaller than that of the computer system itself.”

Greg Nelson and David Redell (1984)

“Since we have no spare planets on which to fight trial nuclear wars, operational testing of a global anti-ballistic missile (ABM) system is impossible.”