The System Usability Process for Network Management Products

by K. D. Gottschalk

This paper presents an overview of a process for system usability. The process is a systematic series of activities and procedures designed to improve the usability of software network management products. The elements of the process are given and future directions for evaluating usability described.

As computer networks become more complex, the difficulty of implementing, operating, and maintaining such networks increases. IBM provides software communications products and network management products to assist in these tasks. If these products can be made more usable by shifting part of the burden introduced by complexity from the system programmer, network operator, and help-desk person to the system itself, a significant growth inhibitor for networks will be removed. The System Usability Process for Network Management Products (hereafter called the process) is a way of ensuring that such a shift occurs. It was developed by the IBM Communication Products Division in Research Triangle Park, North Carolina.

The process described in this paper is an extensive and formal implementation of the general design principles set forth by Gould and Lewis.[1] These principles are as follows:

1. Focus on users and tasks early in the product design cycle.
2. Measure the performance of real users doing real work to judge usability.
3. Do iterative usability testing throughout the design cycle.

Although these principles have been applied by many development groups in many areas (see, for example, the papers by Helander and others[2–5] and the guidelines for software usability design by Rubenstein and Hersh[6]), we believe that our process is somewhat unique both in its scope and in the detail in which it has been defined.

Our process has three major components or aspects:

1. The use of customer surveys and case studies to determine audience and task characteristics for our products
2. The use of measurable usability criteria derived from our case-study data to gauge the usability of our products
3. The use of hands-on testing early in the development cycle to measure the usability of our products

Figure 1 Usability process overview

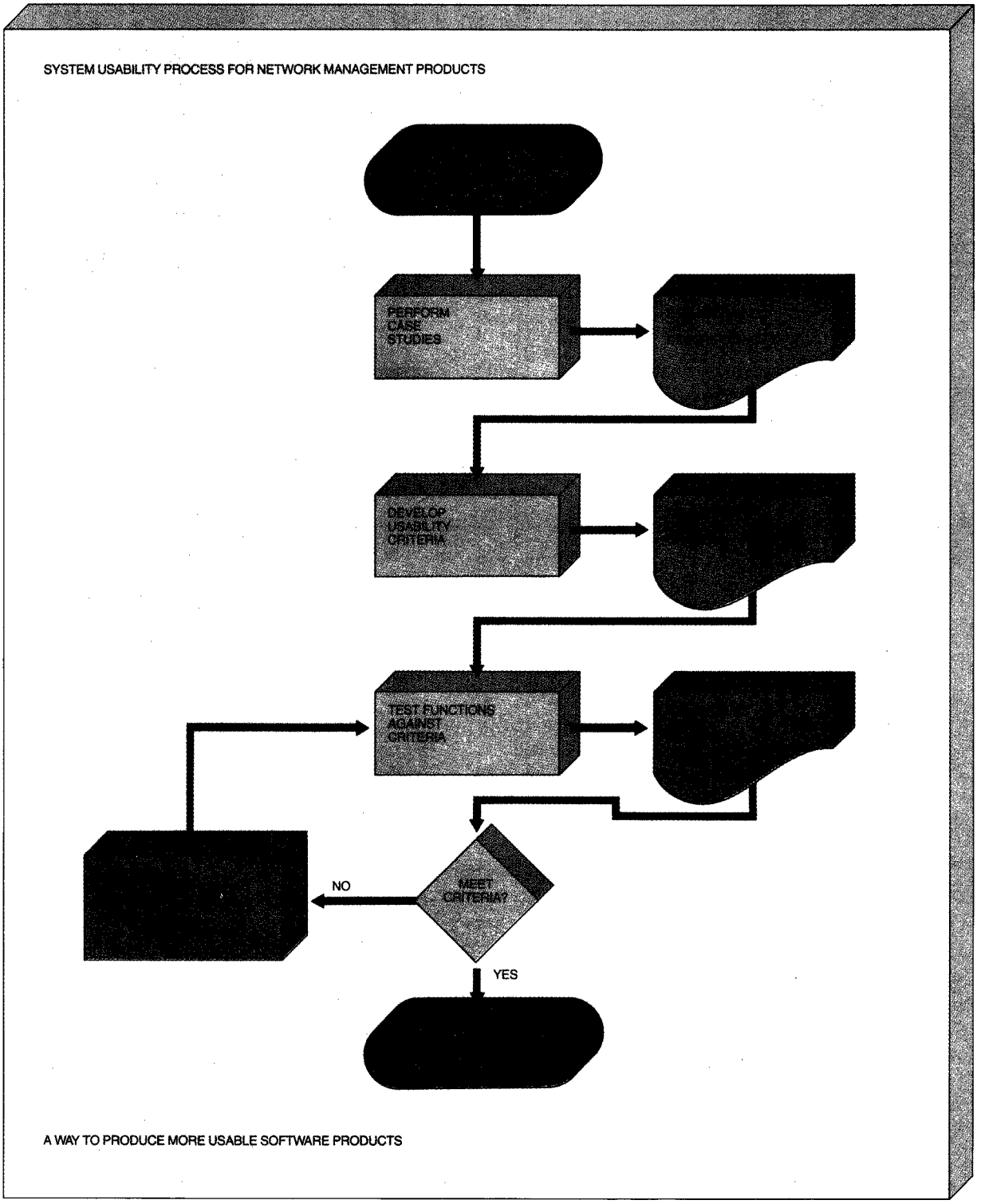


Figure 1 is a flowchart showing how the components of the process fit together. Results of customer surveys and case studies are used to develop product user and task descriptions, from which are derived measurable usability criteria. New product functions are tested against these criteria by having real users perform representative tasks using prototypes of the product interfaces. In areas where the criteria are met, the product is deemed to be usable. In areas where the criteria are not met, the product is deemed to be unusable, and the product interface is reworked and subjected to further testing until the criteria are met.

We now briefly describe each of the major components of the process. Another paper in this issue provides more details on the test component of the process.[7]

Using case studies to determine audience and task characteristics

In 1984 and 1985, a survey was conducted and case studies were performed at 38 IBM customer accounts to determine the characteristics of the users of communications software products and of the tasks that such users perform with these products. Data collected from 488 product users were assimilated into 16 job profiles, which were in turn organized under four categories:

1. Planning for computer networks
2. Building computer networks
3. Operating computer networks
4. Supporting computer networks

Each job profile describes the characteristics (such as experience, education, training, and software familiarity) of the people performing that job, as well as the tasks performed as part of the job. As far as we know, this is the first set of job descriptions in this area that were obtained via extensive surveying and one-on-one interviewing. See the accompanying inset on job profiles of users for an example of one of the job profiles stemming from this study. The full set of job profiles obtained in this study is presented in a report by Beith et al.[8]

As can be seen from Figure 1, the survey and case-study results feed into the next stage of the process, which involves defining usability criteria for future communications software products.

The survey and case-study work described in this issue was very broad; we were attempting to get a general description of the jobs and tasks that exist in a network environment. However, the process is cyclical in nature, since it is performed for each major release of our products. For each such release, new survey and case-study work will be performed. Future survey and case-study work will be based on the work described here but will be more narrowly focused on specific users and tasks, depending on the nature of the products being developed.

Adequate evaluation of the usability of a product requires the use of predefined measurable usability criteria.

Deriving measurable usability criteria

Adequate evaluation of the usability of a product requires the use of predefined measurable usability criteria. As described in another paper in this issue,[7] the criteria we use for usability testing of communications products are expressed in terms of the following variables:

- Percentage of participants successfully completing task
- Percentage of participants having a favorable attitude toward the tools used to perform the task
- Average task completion time
- Maximum task completion time
- Average number of assists
- Maximum number of assists

Prior to each usability test, tasks are identified and criteria are set by the responsible development group, with the assistance of the people responsible for applying the process. Developers and testers

agree upon initial criteria, which are published in a first draft of the test plan. A consensus of all participating groups is reached for the final criteria, which are published in an approved test plan prior to the test.

In our testing we have developed criteria using a number of inputs:

- Experience-based estimates from development planners and marketing people
- The results of previous tests using the same tasks
- Case-study input

In our first tests, we used primarily experience-based estimates to set criteria. Today, however, our criteria are based primarily on previous test results and case-study input.

By developing a set of generic system-level tasks for our products and repeating the set for each product release and for successors to the products, we are in a position to demonstrate improved usability (or lack thereof) over time. Furthermore, we solicit input from the people being tested to validate our criteria.

Case-study input is invaluable in helping us to evaluate and prioritize our criteria. By having a good grasp of the characteristics of the people using our products and the tasks they perform with them, we are in a position to reasonably evaluate the impact of future products and releases on required skill level and worker productivity.

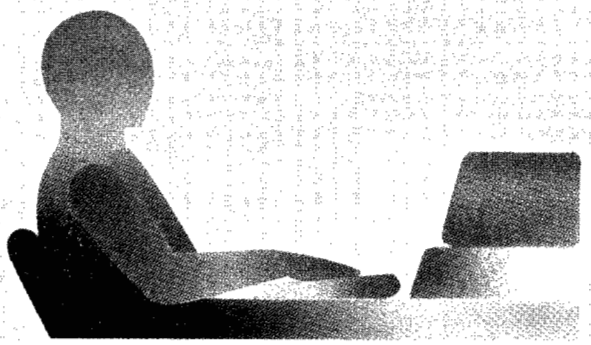
In addition to criteria, we also define problem severities in terms of the relative priority of the task and criterion for which a problem was encountered, and the degree to which the criterion was missed. We work with three levels of problem severity, ranging from mild to severe.

Assessing product usability

Once criteria have been developed, the process requires that a hands-on test be performed to evaluate product usability. The testing process is described elsewhere in this issue.[7]

Tests are done using simulations and real products. Test subjects (whom we call *participants*) are people from IBM customer accounts who perform jobs in which they use our products. Since the testing

Job profiles of users



A survey was performed to provide information on actual and potential users of IBM software communication products in the information systems and data processing industry. The survey was conducted from May 1984 until February 1985 and involved some 38 customers in the United States and the United Kingdom. In total, almost 500 users were surveyed, of whom 328 were also interviewed. The result is a sizable data base of information on users and an initial step toward a comprehensive, standardized set of profiles for users of communications software.

An important part of the usability process is to understand the users for whom products are being developed. This understanding supports the design process by pointing out user characteristics that might affect a user's performance of some task. It also helps us to plan a testing process that allows researchers and developers to establish tasking scenarios and test subjects that have real-world fidelity.

Information for the user profiles came from the following three areas in the survey:

1. Background of a user, including education, software familiarity, extent of experience, major tasks performed, and the scope of his/her job (network, system, or product).
2. Specific work represented by scores that ranked 124 subtasks associated with planning, building, operating, and supporting a network.
3. Work environment with regard to the size of the network in which the user works.

Based on information from 101 respondents, the profile for a systems programmer as one type of communications software user is given below.

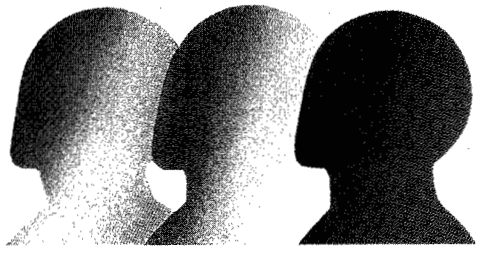
Major tasks. Systems programmers in communications are involved in the task categories of building and operating—heavily so in all the building tasks such as configuring, testing, and programming systems and moderately so in the service task.

Scope of technical environment. Systems programmers are split between network (52 percent) and system (40 percent) environments. Additionally, eight percent work in the product environment.

Experience in current job. Current job experience ranges from less than a year to over ten years; the average is about three years.

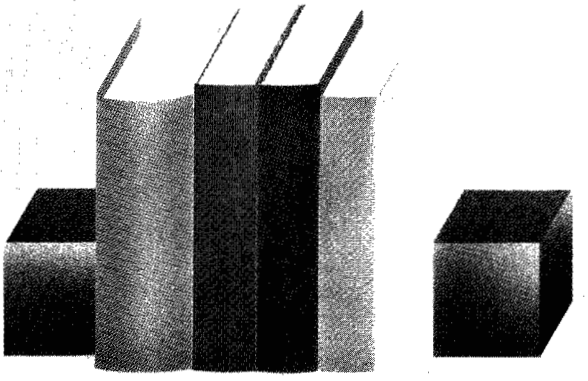
Total experience in technical field. Total experience in the field ranges from none to thirty years; the average is eight years. In their previous jobs, 55 percent of the systems programmers were programmers at some level. No other background group is consistently apparent.

Formal education. Systems programmers are highly educated, with 51 percent having bachelor's degrees or higher and 91 percent having some formal education beyond high school. Of the respondents, 43 percent have no formal education in computer-related areas, whereas 40 percent reported holding a degree or certificate in the field.



Software training. Software training courses have been taken by 84 percent of the systems programmers at some time. Sixty five percent have taken at least one course through IBM. Of all courses taken, 65 percent were taken through IBM. The majority of these courses have covered the Virtual Telecommunications Access Method (VTAM), Multiple Virtual Storage (MVS), Customer Information Control System (CICS), and Network Control Program (NCP).

Software familiarity. Systems programmers are familiar with more products than any other group profiled. In descending order of familiarity by percent, they indicated a knowledge of VTAM–53, CICS–45, Time Sharing Option (TSO)–37, Network Communications Control Facility (NCCF)–35, NCP–34, MVS–29, Network Problem Determination Application (NPDA)–23, Virtual Machine (VM)–23, Interactive System Productivity Facility (ISPF)–21, and Information Management System (IMS)–13. Fifty percent are familiar with four to five of our products, and the maximum known by any systems programmer is 10.



Altogether, after analyzing the survey information, we identified and wrote up 16 job profiles for users of software communications. With these profiles, we are now able to compare and contrast the various users of IBM products and focus on their differences and similarities as important input to the design of a product. The profiles show us which tasks are most important to a specific job type, as well as which jobs share the responsibility for a particular task. This, in turn, is reflected in the requirements for product function, implementation, and interface design.

In addition to systems programmer, the other 15 job profiles are planner, designer, systems analyst, application programmer, help desk operator, computer operator, network operator, maintenance/support, hardware support, and manager. The profile for managers was further broken out into six different jobs depending on the type of group being managed. These six types of managers include planning managers, build managers, operations managers, plan-build managers, build-operate managers, and other managers.

These 16 jobs were then profiled in nine different areas, including major tasks, minor tasks, job scope, current job experience, total experience in the field, previous job, formal education, software training, and software familiarity.

More information on the survey and job profiles may be obtained from Reference 8.

K. D. Gottschalk

is done prior to product announcement, customers sign nondisclosure agreements.

We have performed tests both in the IBM Research Triangle Park Human Factors Engineering laboratory and in an ordinary office environment. The laboratory offers a more controlled environment but tends to make some participants ill at ease. The office setting is not as controlled but results in participants being somewhat more relaxed.

Tests generally are given to four people at a time; these four people are often from the same company. Test runs last one or two days, as it is difficult to bring in customer participants for periods longer than this. A test typically consists of five runs, for a total of 20 participants.

Participants in a two-day test run typically spend the entire first morning in introductory and tutorial activities. For the duration of the run, each participant is assigned two monitors who talk to the participant and try to set him or her at ease. One monitor assigns tasks to the participant and provides help when it is needed. The other monitor is responsible for recording the participant's comments and performance using a program designed for this purpose. During the course of a two-day test run, a participant may perform ten or twelve tasks.

We feel that it is very important that the participant understand the rationale for the test and that the participant be given the option of skipping tasks which for any reason he or she does not wish to perform. We devote considerable time to explaining the rationale behind our process, emphasizing the following points:

1. We are testing the product and not the participant.
2. We do not expect the participant to complete all of the tasks successfully.
3. The participant's individual performance data will not be made available to anyone other than the testers.

Participants' names are not associated with their data.

We feel that it is important to provide each participant with some new knowledge that he or she can take away from the test. Therefore, we provide extensive tutorial information. In addition, if a participant cannot complete a task, the monitors are trained to log this fact (so that we can use it in evaluating the product) but to then provide the participant with sufficient information to allow him or her to complete the task. By silently noting the failure, the monitor provides the information needed to evaluate the product. By providing the participant with the information he or she needs to continue from the point of failure, the monitor helps keep down the participant's frustration level and also provides valuable information that may later be applicable to the participant's job.

Monitors are subjected to an extensive training program. We make heavy use of automated data loggers and trace routines to keep track of individual performance. Performance data are entered into the system in a manner that allows the data to be analyzed easily via automated techniques.

In designing our tests, we try to avoid a sterile laboratory environment. We make sparing use of video cameras and place the monitor in the same room with the participant.

Participants perform the tests individually, but during the final hour or two of a test run we gather participants, testers, and developers together to discuss the overall impressions that the participants have of the usability of the product being tested, and of the test itself. Typically, this portion is the only part of the test that is recorded on videotape. This gathering allows participants to ask questions, vent their frustrations, and develop as a group some consensus as to the major good and bad features of the product with respect to usability. We believe that the group session is a good way for the participants to close out their test experience, and it also allows us to answer any questions they may have and further define potential usability problem areas.

After the test, data are reduced, and a problem evaluation team evaluates the data and assigns levels of problem severity (as described by Percival and Johnson[7]). This team includes members from development, assurance, information development, and human factors areas. Problems are described in detail in usability problem reports, which are given to product developers and are not closed out until the evaluation group is satisfied that either a plan is in place to fix each problem or the

problem description was invalid. (A problem description might be invalid because an inappropriate user class performed the task, because inappropriate conclusions were drawn from the data, etc.) Problem fixes are tested in a later test cycle.

Future directions for the Usability Process

We have developed and refined the System Usability Process for Network Management Products over the course of two years and four separate tests. When we started, we were *certifying* the usability of products that were close to announcement, but we are now beginning to use the process as a *tool* to aid in the design of usable product interfaces very early in the product design cycle. Figure 2 shows how the usability process set forth in Figure 1 is now being used to design more usable interfaces, as well as to certify the validity of such interfaces once they are designed.

We are now beginning to use the process as a *tool* to aid in the design of usable product interfaces very early in the design process.

In the survey and case-study area, we will be performing more specialized and limited case studies of particular tasks and subtasks of interest, in order to supplement the more general information that we gathered in our initial effort.

In the test area, we are attempting to develop tools that will allow us to automatically generate problem severities from test data and are also experimenting with ways of bringing tests to customer locations by running simulations on personal computers. We do not expect testing at customer locations to displace on-site evaluations, but feel that such evaluations can be used in some circumstances to obtain valuable data from people who for one reason or another cannot come to Research Triangle Park to participate in our tests.

Advantages of the process

We believe that the Usability Process for Network Management Products provides us with a flexible and cost-effective way to assess the usability of product interfaces at many points during the product development cycle. The process can be used to certify product usability late in the development cycle, but it can also be used to help develop usable interfaces very early in the cycle.

The process focuses attention on task-related usability problems, and ensures that the problems discovered are the problems that will be encountered by people who normally use these products in doing their jobs. In addition to giving us a way to measure, track, and improve usability, the process allows us to bring the product user into the product design cycle at a very early point. Process features which we believe to be unique are as follows:

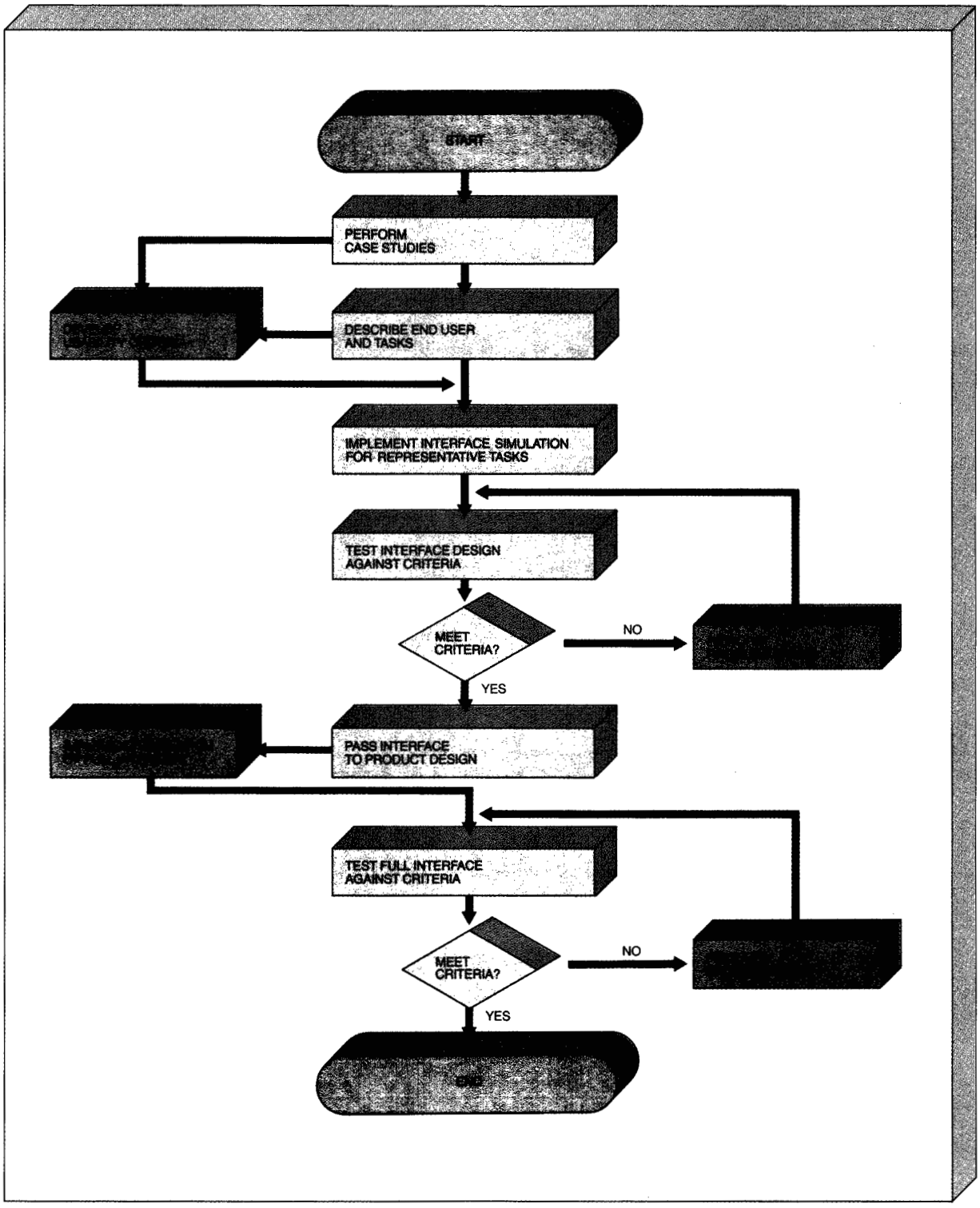
1. Extensive and detailed survey and case-study work with real users of network management products
2. Development of usability criteria based on an extensive amount of real-world data
3. Applicability of the process in all its detail to many different points in the product development cycle

We are excited about our process and believe that it will lead to significant improvements in the usability of our communications products.

Cited references

1. J. D. Gould and C. Lewis, "Designing for usability: Key principles and what designers think," *Communications of the ACM* **28**, No. 3, 300–311 (March 1985).
2. G. A. Helander, "Improving system usability for business professionals," *IBM Systems Journal* **20**, No. 3, 294–305 (1981).
3. K. A. Butler, "Connecting theory and practice: A case study of achieving usability goals," *CHI '85 Proceedings*, 85–88 (April 1985).
4. C. B. Mills and K. L. Dye, "Usability testing: User reviews," *Technical Communication* **32**, No. 4, 40–45 (1985).
5. M. A. Lund, "Evaluating the user interface: The candid camera approach," *CHI '85 Proceedings*, 107–113 (April 1985).

Figure 2 Process extended to interface design validation



6. R. Rubenstein and H. Hersh, *The Human Factor: Designing Computer Systems for People*, Digital Press, Burlington, MA (1984).
7. L. C. Percival and S. K. Johnson, "Network management software usability test design and implementation," *IBM Systems Journal* **25**, No. 1, 92–104 (1986, this issue).
8. B. H. Beith, G. C. Moore, W. L. Pendley, and L. C. Percival, *Users of IBM Communication Software: A Survey of Customer Jobs, Tasks, and Networks*, IBM Corporation, P.O. Box 12195, Research Triangle Park, NC 27709 (1986); available from authors.

Karl D. Gottschalk *IBM Communication Products Division, P.O. Box 12195, Research Triangle Park, North Carolina 27709.*
Mr. Gottschalk is a senior programmer in the Network Management Advanced Design group in Raleigh. He joined IBM as an associate writer in 1968, working on the OS/TCAM library. Since then he has held a variety of planning, programming, writing, and testing assignments dealing with communications software. He is currently involved in the design of more usable user interfaces for network management software products. Mr. Gottschalk received an M.A. in English literature from the University of Mississippi in 1965, an M.S. in computer science from the University of North Carolina in 1973, and an M.B.A. from Duke University in 1983.

Reprint Order No. G321-5263.