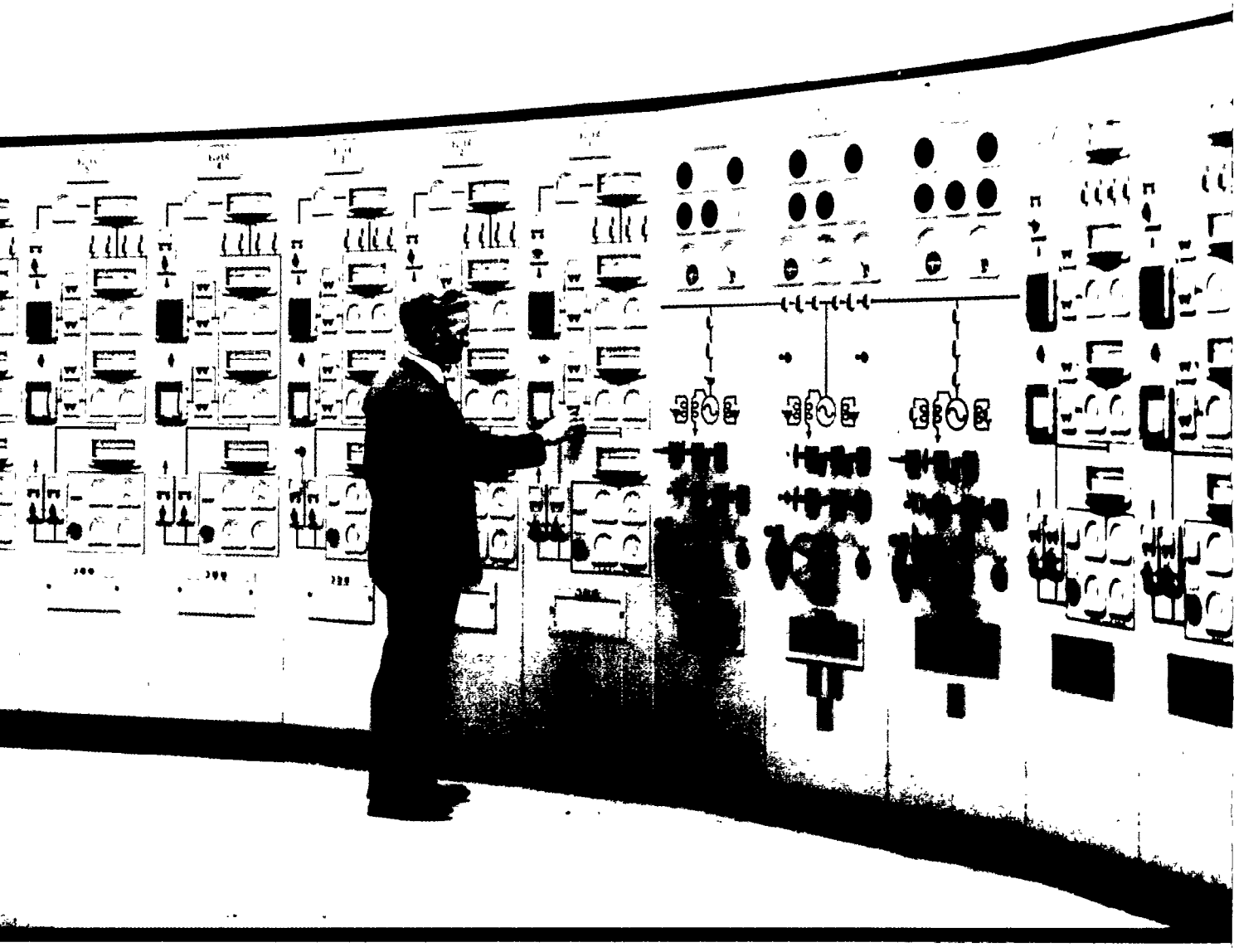
Bill Curtis
Bill Hefley

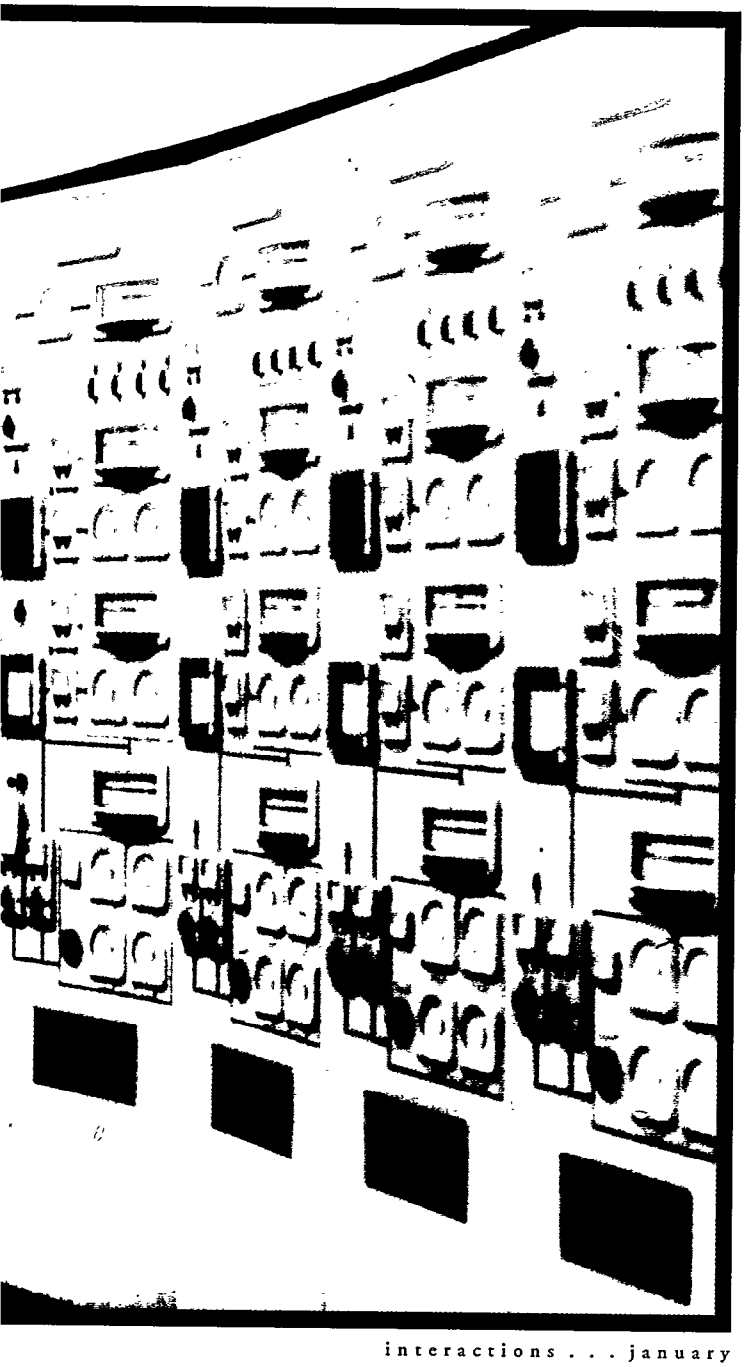
A WIMP No More

The Maturing of User Interface Engineering



Archive Photos/London Times

Executive—I'll worry about the user interface when someone can demonstrate that it makes a difference in my sales
Project manager—Yes, I'm sure you would love to do some field testing with users, but there's no slack in the schedule or budget
System designer—That's trivial, let them handle it in the user interface
Software engineer—When I'm done they have somebody who comes around and makes the screen look pretty
Interface engineer—Isn't it exciting, I get to design the user interface all by myself
Customer—We require that any software we buy have a GUI, you know, a Generic User Interface



In earlier times human factors specialists were called in to look at a control panel and decide how the knobs and dials should be arranged. When computer terminals became a common interface to systems, user interface specialists were called in to design the screen layout and dialogue. Now software is controlling products and services ranging from home entertainment to banking to air traffic control systems and medical devices. Consequently, there is greater demand for engineering discipline in the design of user interfaces.

Every quote above represents an attitude about user interface engineering that emanates from the particular responsibilities of the speakers position. Attitudes like these have kept user interface designers from playing their full role in systems development. The traditional attitudes toward interface development

About the Authors

BILL CURTIS

Bill Curtis works half-time with the Software Engineering Institute at Carnegie Mellon University, is a founding faculty member of the Software Quality Institute at the University of Texas, and works with organizations to increase their software development capability.
email: *bcurtis@cs.utexas.edu*

BILL HEFLEY

Bill Hefley is affiliated with the Department of Engineering and Public Policy and the Software Engineering Institute, both at Carnegie Mellon University. His experience is in Systems Engineering, Human Factors Engineering and Software Development functions.
email: *weh@sei.cmu.edu*

represented in these quotes are slowly changing, and the pace of change will quicken in the next decade.

These changes signify that user interface development is being accepted as an equal partner in systems development. Yet they also require that user interface designers will be assimilated into the systems engineering process. While user interface designers will have to embrace new disciplines, they will contribute skills being increasingly demanded in systems development. They will bring a keen understanding of how to design systems around users [8, 21]—a skill too long missing from the teams that develop most computer-based products.

The practice of user interface design is barely two decades old. Only since the early 1980s have user interface designers been identified as a separate skill in systems development. User interface practitioners and scientists first met to form themselves into a field in Gaithersburg, Maryland, in March of 1982 at the first of what has become the annual CHI conferences. During the 1980s the practitioners of this specialty worked to develop their skills into a discipline. It is still a young discipline, but the demands confronting it in the 1990s are bringing its childhood to an end.

User interface designers will not undergo these changes alone. Software engineers are feeling the same tremors. Skills such as software design and interface design were once practiced separately and all too often met only at system test. Computer-based systems can no longer be built that way. There are four global trends driving the changes user interface designers are about to experience.

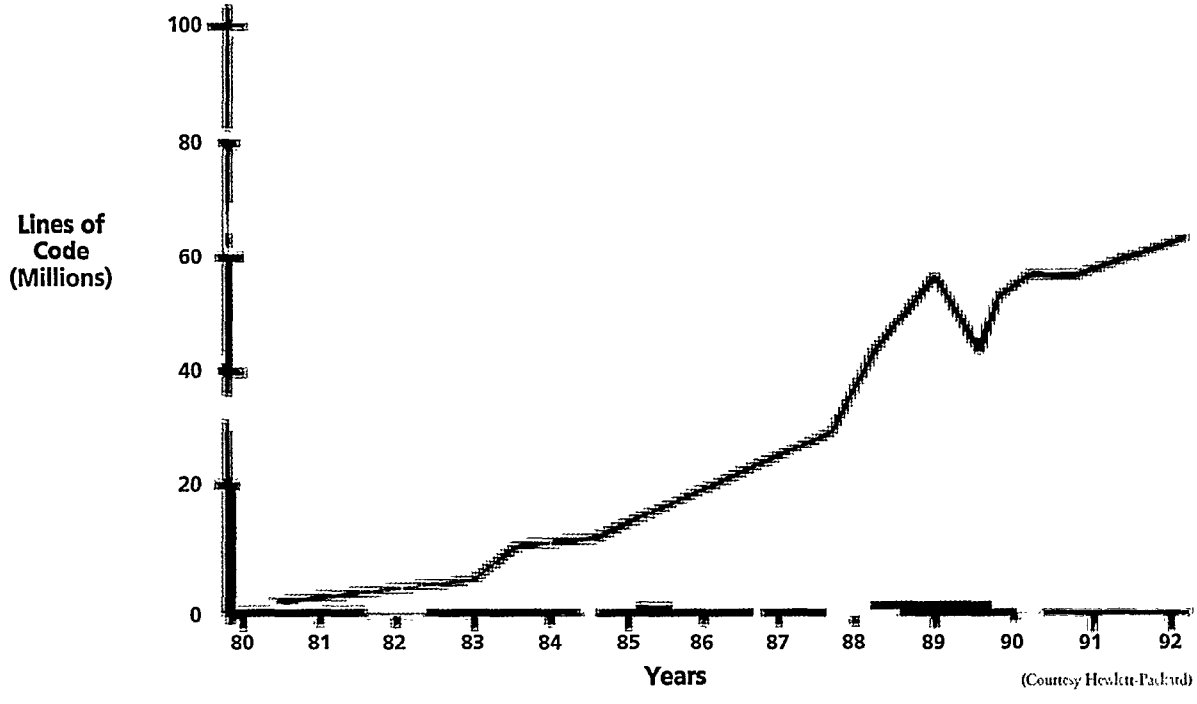
Four Trends Reshaping User Interface Practice

The software explosion. Forget the "software crisis," it's now a software explosion. This explosion is driven both by customers who demand more functionality from systems and by the endless freefall in the cost of MIPS. Even with less expensive hardware, many of these burgeoning requirements are cheaper to satisfy in software. Much of this growing functionality will be put under the control of end users, users already overcome by the interface to their VCR.

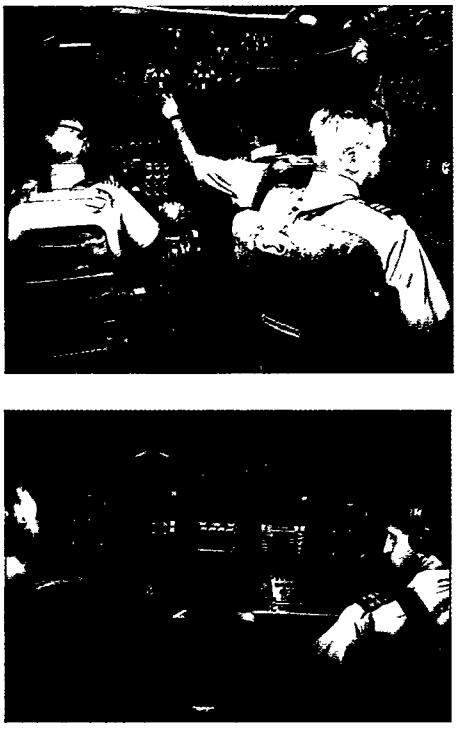
As a result of this explosion, software is accounting for a larger proportion of the effort and cost of building most systems. In fact, the software inside most products and systems is growing by an order of magnitude each decade, and sometimes each half decade. In his keynote

Figure 1

Growth of Software in Hewlett-Packard's product lines



(Courtesy Hewlett-Packard)



(Courtesy NASA - Ames Research Center)

Figure 2
Cockpit Instrumentation in B727 (top) and Advanced Concepts (bottom) flight simulators

address at the Software Engineering Institute's 1993 Software Engineering Symposium, Lewis Platt, Chief Executive Officer of Hewlett Packard, described the growth of software in HP's products [23]. Figure 1 represents this growth of software across all of HP's products. In its initial laser printer product, HP had 25,000 lines of code; today, that number has grown to 200,000 lines of code; and next-generation laser printer products that will appear on the market in about 18 months will have one million lines of code. These data are typical of products across industry.

Over several generations of a product line, user interfaces can potentially become exponentially more complex. Although many functions formerly performed by the user have become automated (e.g., fly-by-wire avionics), the access to information and the decisions to be made about increasingly complex systems grows dramatically. The examples of cockpit instrumentation shown in Figure 2 demonstrate how this interface has changed, and with it the job performed by the pilot changed as well (from steering to flight management). The challenge to interface engineering is to devise clever and efficient interfaces that make new systems seem easier to access, learn, and use than the previous generation.

Catherine Plaisant and Ben Shneiderman [22] have demonstrated that the more esoteric problems of setting a VCR could be tamed by a good WIMP (windows, icons, menus, and pointers) interface (Figure 3). The VCR problem explodes in complexity when the next generation TV becomes the interface to at least the home entertainment system, and possibly to the home management system, home shopping and banking systems, and Vice President Gore's Information Highway. The use of these systems will be limited by the user's ability to navigate and steer them. A better rudder will not be enough; interface designers must do for our travelers of virtual space what global positioning systems, radar, and wireless communications have done for seafarers.

The communications explosion. Two decades of effort to integrate computing and communications is creating a revolution in user interface work. The core of user interface work used to be the interface for a single user. Yet as businesses move more toward team-based processes, and as communication technology provides universal connectivity through all manner of media, the core of interface work will expand to include, and perhaps even focus on multiuser interfaces.

The field of computer-supported coopera-

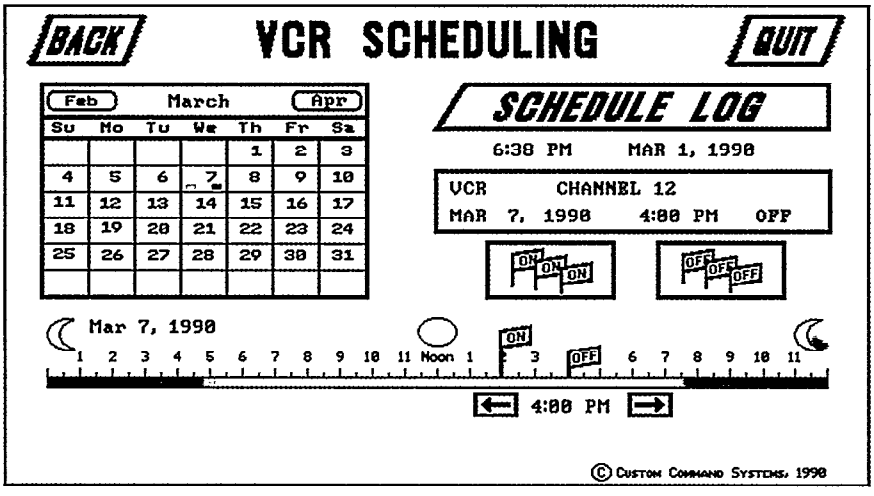


Figure 3
Home automation system on a common personal computer, shown scheduling a VCR.

(From Plaisant & Shneiderman, 1991) (Courtesy Custom Command Systems Inc. and University of Maryland)

tive work (CSCW) also developed during the 1980s [9], but with fewer practitioners than were designing interfaces for single user programs. With fewer experienced designers available, the growth of communication capabilities will far outstrip our ability to develop effective multiuser interfaces for them. Even now most email systems have user interfaces that were designed for mainframe-based terminals 20 years ago. Few using email systems have avoided the embarrassment of broadcasting a private message to an entire mailing list because they inadvertently chose the wrong reply mode. The interface capabilities required to allow people to work face to face from remote locations is more complex because of the synchronization requirements [5]. For instance, how do we control the versioning of a document we are working on concurrently?

The rapid expansion of communication services is creating a rapid expansion in the variety of people who use computer-based applications. The interface experience base for many of these new users will have been limited to the VCR and microwave oven, and in both cases the clock is still flashing 12:00. The debate over universal interface standards versus creating leeway for interface innovation will escalate. In either case the growing diversity of users reached through expanding communication capabilities will challenge our ability to provide interfaces that make the services easily accessible to virtually everyone. Consider the issues of literacy, bilingualism, visual acuity, aging, special needs, multiculturalism, technophobia, and other human attributes that must be accommodated. The challenge of expanding the user population beyond the those who ordinarily use computers should lead to innovations in interface design.

The media explosion. The media through which users interact with systems have expanded dramatically over the last decade. Three-dimensional mice, pens, touch screens, video, speech, helmet-mounted displays, virtual environments, and other interface capabilities have increasingly burdened software designers to integrate input and output through different devices. Managing this jungle of I/O requirements in addition to the growing size of application dialogues has driven the software devoted to user interface functions to half the code in most current systems [18]. Thus, the user interface has become a core systems-engineering issue separate from its usability concerns.

Not only is the amount of systems software devoted to user interface management growing, but its complexity grows as the range of I/O devices grows. The control and integration of dialogues conducting through these different I/O channels will require innovative interface control architectures such as those explored at MCC and elsewhere [10, 12, 16].

The diversity of I/O channels available with many systems will challenge user interface designers to determine which combinations of I/O channels and dialogue types are most usable for which applications. Similarly, products hyped on capabilities, such as handwriting, speech, or gesture recognition, that do not achieve exceptional recognition accuracy will find their sales life short when competing with products that offer less sophisticated, but more reliable interactions. For instance, a VCR remote control that uses voice recognition may still have to compete with traditional push-button remote controllers because:

- even with training recognition accuracy may not be high enough,
- the speaker's voice interferes with hearing the TV,
- fingers may be quicker for frequently used commands, and
- names of less frequently used commands may be hard to remember.

Products will compete in the marketplace not only on their interface media, but also on the comparative usability and accuracy of the specific media they employ.

The usability explosion. Usability is driving the marketplace. The potential user population, for which these usable interfaces must be designed, is growing to encompass everyone [19]. However, what the purchasers of systems consider the prime usability issues are not necessarily the same issues that user interface designers normally address. To users, the primary issue in usability is, not surprisingly, whether they can use it. Most user interface

Books and Journals

In addition to the archival literature and the annual CHI conferences, an increasing number of books have been published in recent years which address an understanding of users and methods and techniques necessary to design, develop and evaluate usable systems.

Understanding the User

Card, S. K., Moran, T. P., and Newell, A. The Psychology of Human-Computer Interaction. Hillsdale, N.J.; Lawrence Erlbaum, 1983.

Norman, D. A. The Psychology of Everyday Things. Basic Books, New York, 1988. (Later versions released as The Design of Everyday Things.)

Norman, D.A. Turn Signals are the Facial Expressions of Automobiles. Addison-Wesley, Reading, Mass., 1993.

Norman, D.A. Things That Make Us Smart. Addison-Wesley, Reading Mass., 1993.

Methods and Processes

Bias, R.G. and Mayhew, D.J. Cost-Justifying Usability. Academic Press, Boston. In press.

Card, S. K., Moran, T. P., and Newell, A. The Psychology of Human-Computer Interaction. Lawrence Erlbaum, Hillsdale, N.J., 1988.

Cox, K. and Walker, D. User Interface Design 2nd ed. Prentice Hall New York, 1993.

Diaper, D. Task Analysis for Human-Computer Interaction. Ellis Horwood Limited, Chichester, U.K., 1989.

Dix, A., Finlay, J., Abowd, G., & Beale, R. Human-Computer Interaction. Prentice Hall, New York, 1993.

Greenbaum, J. and Kyng, M. Design at Work: Cooperative Design of Computer Systems. Lawrence Erlbaum Associates, Hillsdale, N.J., 1991.

Hix, D. and Hartson, R. Developing User Interfaces. Wiley, New York, 1993.

Helander, M. Ed.. Handbook of Human-Computer Interaction. North-Holland, Amsterdam, 1988.

Johnson, P. Human-Computer Interaction. McGraw-Hill, Maidenhead, Berkshire, U.K., 1992.

Meister, D. Behavioral Analysis and Measurement Methods. Wiley, New York, 1985.

Nielsen, J. Usability Engineering. Academic Press, Boston, 1993.

Malin, J. T., Schreckenghost, D. L., Woods, D. D., Potter, S. S., Johannesen, L., Holloway, M., and Forbus, K. D. Making Intelligent Systems Team Players: Case Studies and Design Issues, Vol.1: Human-Computer Interaction Design (NASA Technical Memorandum 104738). NASA Johnson Space Center, Houston, Tex., 1991.

Norman, D. A. and Draper, S. W. Eds. User Centered Systems Design: New Perspectives on Human-Computer Interaction. Lawrence Erlbaum, Hillsdale, N.J., 1986.

Rubenstein, R. and Hersh, H. The Human Factor: Designing Computer Systems for People. Digital Press, Maynard, Mass., 1984.

Shneiderman, B. Designing the User Interface: Strategies for Effective Human Computer Interaction 2nd ed. Addison-Wesley, Reading, Mass., 1992.

Thimbleby, H. User Interface Design. ACM Press, New York, 1990.

Journals on User Interfaces

As the user interface discipline evolves, a number of journals have begun which focus on this important area. These journals include:

- ACM Transactions on Computer-Human Interaction (ACM)
- Human-Computer Interaction (Lawrence Erlbaum Associates)
- International Journal of Man-Machine Studies (Academic Press)
- Behavior and Information Technology (Taylor and Francis)
- Interacting with Computers (Butterworth Heinemann)
- ACM Transactions on Office Information Systems (ACM)
- International Journal of Human-Computer Interaction (Ablex)
- User Modeling And User-Adapted Interaction (Kluwer)

The biggest usability explosion, and one that user interface designers could do a better job of accelerating, will come when businesses are able to account for the real costs of poor user interfaces.

designers interpret this to involve effective screen layout, uncomplicated dialogue, and mnemonic icons. Since interface designers narrow their interpretation of usability to attributes of the user interface, they DO NOT fully understand what motivates their users.

To users who were tired of being enslaved by a single vendor, usability first means that an application is available for use on their machine. An open architecture is the first and greatest usability issue, because a closed architecture means some applications cannot be used. Availability is an absolute limiting condition for usability. Thus, an easier-to-use interface may not gain market share if substantially more applications can be used on a competitor's offerings. User interface designers are often surprised when they discover which attributes make a system usable to potential customers.

The biggest usability explosion, and one that user interface designers could do a better job of accelerating, will come when businesses are able to account for the real costs of poor user interfaces. For instance, many companies employ hundreds, perhaps thousands of people who perform repetitive tasks at workstations while interacting with customers. These jobs include telephone operators, sales clerks, and stock brokers. The benefits of better interfaces for these jobs can range from the hundreds of thousands to millions of dollars for a single company. These benefits can be realized through:

- reduced system interaction time per customer interaction,
- reduced learning time for new hires or assignees,
- fewer mistakes requiring additional customer interaction, and
- improved customer service that enhances repeat business.

In an early example of this type of analysis, the comparative interaction times of similar functions were analyzed on two different operator workstations. The cost differential for a 0.8 second difference in each transaction spread across all of their operators accounted for $2,400,000.00 in savings per year [7]. This difference means that the same work can be done with fewer people and the savings passed directly to the bottom line.

The user interface community needs to begin looking for opportunities to demonstrate dramatic cost reductions [2]. As the impact of usability on costs becomes clear in the executive mind, there will be an explosive demand for comparative usability studies that will reward those vendors offering the most effective user interfaces. When this happens, the usability issues understood by most user interface specialists will become decisive issues in the market. We are still some years away from this general awareness of the business value of usability.

Becoming a Partner

These four global trends will accelerate the pace at which user interface design is assimilated as a primary contributor to the systems analysis and development process. Although interface engineers have always wanted to be an integral part of the systems development team, they are usually called in after systems analysis has been performed. Today the requirements for most systems are typically analyzed into

those that will handled in hardware and those that will be executed in software. Any requirements not formally assigned in this systems analysis process are implicitly assigned to users. Only a few applications such as avionics and a limited number of other life-critical systems incorporate a formal assignment of requirements (or tasks) to system operators.

Integrating the disciplines. When most of the people analyzing a system have computer science or electrical engineering degrees, the focus is usually on the technology. New approaches to system development such as user-centered system design [21], participatory design [8], or Scandinavian design [3, 6] that made the end user rather than the technology the focus of the design process were proposed in the 1980s. Although still not widely adopted outside of Northern Europe, the concepts underlying these approaches are starting to appear in other system development movements.

New approaches to systems development such as concurrent engineering and cross-functional development teams are starting to gain momentum in the United States. These interdisciplinary approaches integrate different design responsibilities earlier in the development process in order to avoid expensive redesigns downstream. The objective is to identify dependencies among different system components earlier and manage their integrated development more effectively.

User interface engineers always believed that "interdisciplinary" meant that the other guys had to get used to us. However, as user interface designers assimilate into the system development process, they must accept new responsibilities. Primary among these changes will be to make user interface design and development an engineering discipline and to integrate interface-engineering processes with those of other engineering disciplines. The changes in the system development process occurring in the 1990s will provide equal discomfort for all technicians who had hoped to practice their discipline in solitude.

Developing engineering discipline. The popular image of user interface design is of psychologists and graphic designers engaging in a creative, artistic process. While there is truth to this image, there is much more behind successful user interface design than the exercise of creative artistry. Methods exist for task analysis, user modeling, usability specification, participatory design, dialogue complexity analysis, interaction analysis, empirical testing in usability labs, and contextual analysis in the field, among others [11]. Most of these methods have been evaluated empirically, and this research has been reported in the archival literature and at the annual CHI conferences.

What has been slower in developing is the formal recognition that there exists the methodological basis for both an academic discipline and an engineering practice in user interface development. Industry has struggled with understanding that software engineering requires a disciplined process. Being younger and less grounded in either mathematics or physics, user interface design faces even greater hurdles. Yet system developers are facing challenges, both legal and competitive, that require user interface development to become more rigorous.

For instance, in analyzing the software faults that led to fatal doses of radiation being delivered to several patients by the Therac-25, Leveson and Turner [15] identified how at least some of the deaths were caused by an interaction of user behavior (typing speed) and module design. The analysis of the interaction of the user and the system, especially in life- or business- critical applications, must be disciplined and thorough in order to avoid disastrous results. As vendors are increasingly held legally responsible for the performance of their systems, greater pressure will be placed on establishing a user interface engineering discipline that can analyze the impact of user behaviors in the context of system function and design choices.

The user interface engineering discipline needed in industry must be grounded in rigorous training provided by the universities. Even though the Special Interest Group on Human-Computer Interaction (SIGCHI) of the Association for Computing Machinery has produced a proposed Curricula for Human-Computer Interaction [1], only a handful of universities have allowed students to take such a course of studies as a major field of special-

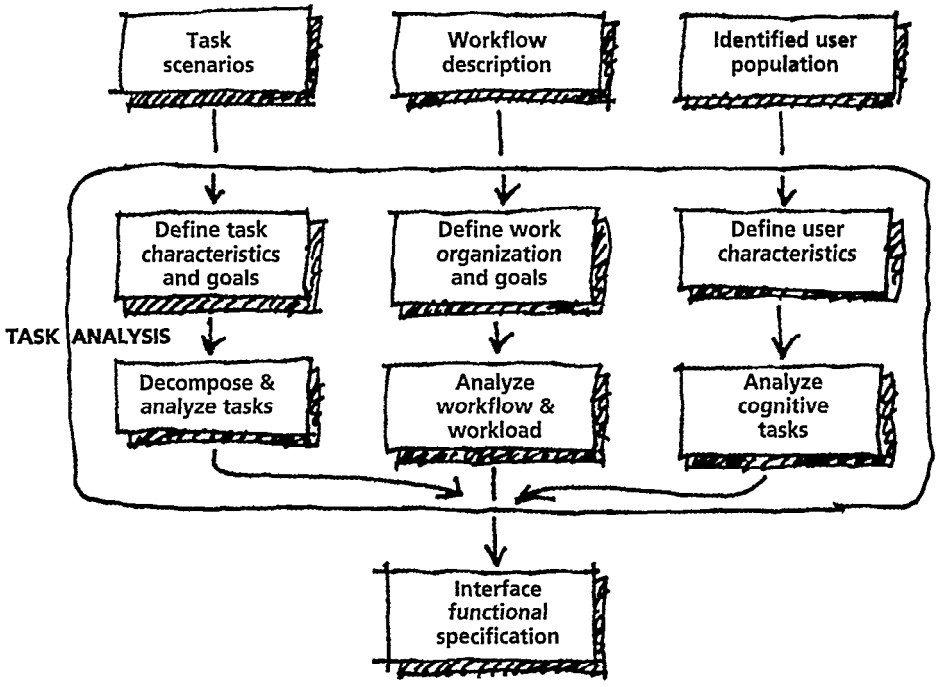


Figure 4
Initial decomposition of the task analysis component for an user interface engineering process architecture

ization in either psychology, computer science, or engineering. Not surprisingly, during the 1980s assistant professors who chose this as their primary field of research found difficulty gaining tenure in most research universities.

The User Interface Engineering Process

There are three requirements for integrating user interface engineering into product engineering. First, a process needs to be defined for specifying, designing, building, testing, and evaluating a user interface. Second, this defined process needs to be integrated with the defined process used for developing the remainder of the product (hardware, software, etc.). Third, the organization must have an established project management discipline, so that it can manage a well- defined process and avoid making commitments that even a sound engineering process could not satisfy. In the face of unachievable commitments, most project teams circumvent their defined process, usually with embarrassing results.

Defining the process. Defining a process for interface engineering communicates to everyone in the organization what interface professionals do and how these activities integrate with the rest of a project's engineering activities. When user interface activities are not defined, managers often misunderstand the resources needed, how interface work should be integrated with the rest of product engineering, and how to track progress in user interface development. If a company already has a well-articulated development process, then the process for developing user interfaces must be described within its constraints. An early example of integrating interface engineering into an existing system development process was described by Larry McLaughlin [17] at TRW. However, if a company does not have a well-defined development process, and most do not [14], then there is an excellent opportunity to participate in developing one that includes user interface development as an integral component of product engineering.

In order to define a user interface engineering process, the major stages involved in developing an interface (e.g., user and task characterization, design, implementation, user testing, etc.) must initially be described abstractly. In one such model, Jakob Nielsen [20] described these stages as:

Predesign stage
- Know the user
- Competitive analysis
- Setting usability goals

Design stage
- Participatory design
- Coordinated design
- Guidelines and heuristic analysis
- Prototyping
- Empirical testing
- Iterative design

Postdesign stage
- Collect feedback from field use

The explicit activities and responsibilities of each stage must then be described in detail. Figure 4 displays a high-level decomposition of the task analysis component of a defined user interface engineering process. The next level of detail in this process description would indicate which groups provide input to these tasks, how these inputs are processed, and who consumes their outputs.

As another example, the defined process will indicate a requirement to integrate activities for testing or evaluating the user interface. The detailed process description will integrate activities such as selecting evaluation methods, reviewing the coverage of test cases or usage scenarios, executing tests or evaluations, coordinating changes with the software team, ensuring changes are updated in training and user manuals, and other activities. Although methods for testing or evaluating an interface may differ qualitatively from methods employed in software engineering, they should nevertheless be planned and managed as an engineering discipline.

Integrating defined processes. Having defined a detailed interface development process, the next requirement is to map it into the organization's product development process. Figure 5 displays how this mapping might occur in the context of a traditional waterfall product development model. Since the specific activities of user interface engineering will differ between flight simulators and spreadsheet products, different organizations and different product lines within organizations may have substantially different interface development process models.

Many questions must be answered in a well-defined process. For instance, who develops the product requirements allocated to the user interface? How are these requirements balanced against the abilities of the user population and the typical task profile? How is the relevant population of potential users determined? Which disciplines should be involved in designing the interface? At what point should different skills be inserted into the design process? When will reviews be held and how will a review team be selected? What is the schedule for integrating interface components into the product build and test sequence? How and when will developers of user manuals and training materials be integrated into the process? Who is responsible for handling legal issues related to the intellectual property represented in the interface? Who makes decisions about changes to the interface based on field experience and problem reports?

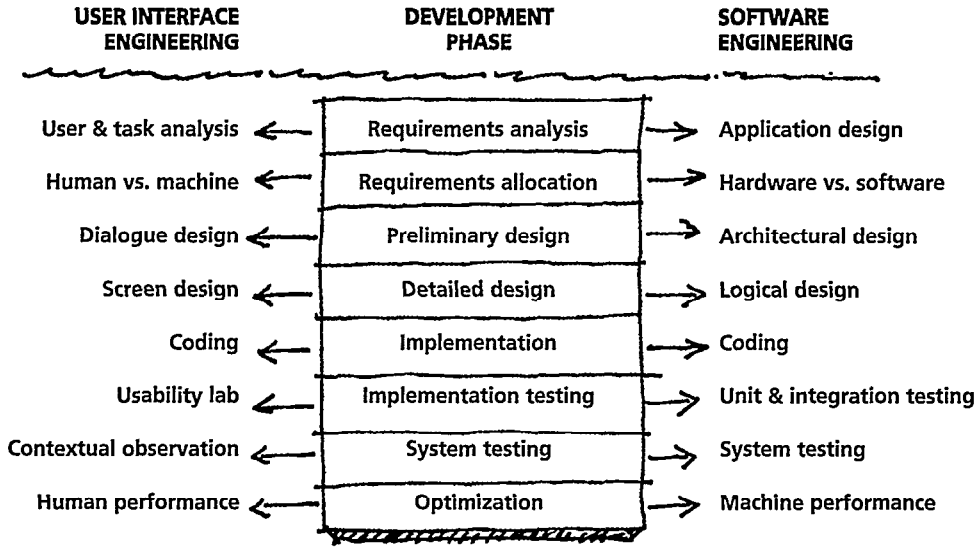


Figure 5
Integrating User Interface Engineering Processes

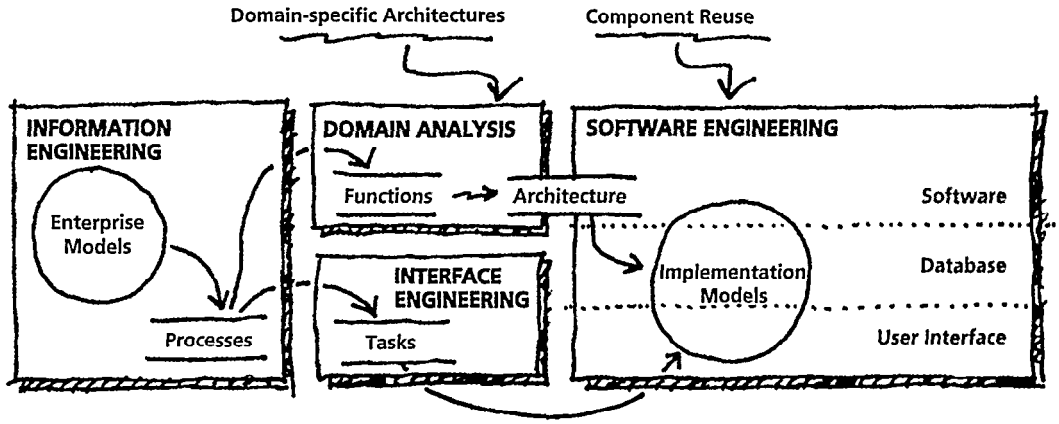


Figure 6
Comprehensive Systems Development Process Architecture

It is this crucial step of mapping the interface development process into the rest of the product-engineering process that embeds user interface engineers into the product-engineering team. Describing the activities of user interface development this carefully also demonstrates that interface development is transitioning from an artistic exercise into an engineering discipline. It provides visible evidence that the "soft sciences" can produce rigorous methods for engineering solutions to practical problems.

Managing user interface engineering. The final requirement for institutionalizing an interface development process is that the organization has established a project management discipline. Defining the interface development process is valuable because it communicates to the organization the methods and requirements for interface engineering and how these relate to the larger process of developing a product. However, without management support, the advantages of following a defined process can easily be sacrificed. For instance, if management commits to unrealistically short schedules for product delivery, many required components of a sound engineering process will be curtailed to meet the deadline. Often interface design methods, reviews, and user testing are the first processes to be sacrificed.

To lay the foundation for institutionalizing a sound engineering practice, an organization needs to develop enough management discipline to ensure that product baselines are established and controlled, requirements changes are tracked and controlled; resources are adequately estimated; project activities are planned and tracked; and process quality is assured. Without this discipline, sound engineering can be undermined by uncontrolled commitments and poor management. Too many user interface designers have been become accustomed to being called in at the last minute to fix a grotesque interface shortly before delivery. There is little opportunity on such projects to perform creative design. The objective is to save the patient.

When management can establish a controlled environment for product development, user interface engineers can expect to follow a defined process without being forced to circumvent their professional methods. Far from restricting an engineer's design creativity, such environments protect the time an engineer needs for exploring creative design alternatives.

In a disciplined environment, the value of a defined interface process stretches beyond communication and provides the best assurance of producing a user interface that satisfies the needs and expectations of the customer. Undisciplined, schedule-driven projects not only risk missing the schedule, but missing the customer as well. A defined interface development process must be nourished by a commitment to process discipline throughout all aspects of system development.

The Future

The development of computer-based systems is changing, and these changes will ripple through the user interface world. During the 1990s, user interface development will transition from a specialty into an engineering discipline. User interface professionals will find

themselves becoming a more important part of the development team, but they will also find that this requires greater discipline in their work. This discipline is not just technical, it also involves management accountability.

Just as software engineering is becoming a discipline with a defined, managed process [13], user interface engineering will evolve to a discipline having its own defined interface development processes which are practiced by people from numerous fields employing a rich catalog of analytic, design and development techniques. To accommodate these changes, managers who are now struggling to understand software development will quickly have to come to grips with user interface engineering, a discipline even more alien to their experience base than software engineering. Software engineers will have to accept greater input into their design process based on consideration of the user. User interface engineers will have to not only adjust to being a defined part of the systems development process, but they will have to assume greater responsibility for representing users and their behavior during the systems development process.

Since one of their special skills is the understanding of user behavior, user interface engineers should accept special responsibility for being the user's proxy in the development process. They must think behind the requirements to what users want, are capable of doing, or may try to do. User interface engineers must always be asking, "Yeah, but what if the user did this real fast and then got confused and tried to get out this way, unlikely as that may be?"

User interface engineers will continue to have an increasing influence on the systems development process. Figure 6 represents a macrolevel perspective building on classical information systems design to acknowledge: the central role of (1) domain analysis and reusable/tailorable domain-specific software architectures and (2) interface engineering [4] to define operable interfaces which are implemented in software, and the driving role of information engineering (or business process design or business process reengineering) to define the enterprise model describing the functions of the enterprise.

Systems developers will be increasingly concerned with building systems that are operable by their users, interoperable, and enable new and enhanced business functions rather than just continuing to automate the same functions.

Incorporating User Interface Engineering

In an era when software engineering is debating the possibility and merits of "zero-defect software" user interface engineers should challenge themselves to eliminate "operator error" from the lexicon. Just the debate over what operator errors are and what causes them will stimulate most systems developers to start thinking "outside their box" about how to design the best systems for their users. Further, by providing this perspective, user interface engineers can contribute to reducing the risks of product liability.

The way we think about serving users will also change. Considering the generational differences in computer-based systems depicted in Table 1 and the current emphasis on reinventing business processes, user interface engineering will be able to make major contributions to understanding how work is performed within organizations. Although we will always consider the single user at an interface, we will increasingly think about multiple users interacting in time and space. As local area and wide area networks continue to connect more of the world, computer- based systems will continue to move from the mainframe-based transaction model to workstation-based interaction models and on into social and organizational-based interconnection models. The current interest in team-based computing (e.g., CSCW) is only a

Table 1

Generational Differences in Computer-Based Systems

generation	driver	focus
Paleozoic	Technology	Hardware
Medieval	Information Domain	Databases Functions
Renaissance	Organization User	Processes Interfaces
Aquarian	Communities Collectives	Communication Integration

forerunner of the need to provide interfaces for systems that support organization-based and, ultimately, community-based computing.

Over the next decade user interface work will change dramatically. It will become an engineering discipline. It will become a defined part of the systems engineering process. Topics that have been on the periphery will increasingly become core issues (e.g., workgroup interfaces). User interface professionals will regularly compete for executive positions in corporations, and they will begin winning more often. They will win because they will be seen as understanding the customer better in an environment where purchases are increasingly user driven. All new engineering disciplines struggle for respect. The 1990s will be the decade user interface engineering professionals earned theirs. ■

Acknowledgments

This work is sponsored by the U.S. Department of Defense. The views and conclusions contained in this document are solely those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the Software Engineering Institute, Carnegie Mellon University, the U.S. Air Force, the Department of Defense, or the U.S. Government.

References

[1] ACM Special Interest Group on Computer-Human Interaction Curriculum Development Group ACM SIGCHI Curricula for Human-Computer Interaction. ACM, New York, 1992.

[2] Bias, R.G. and Mayhew, D.J. Cost-Justifying Usability. Academic Press, Boston. In press.

[3] Bødker, S. Through the interface-A Human Activity Approach to User Interface Design. Lawrence Erlbaum Associates, Hillsdale, N.J., 1990.

[4] Curtis, B. and Hefley, B. Defining a place for interface engineering. IEEE Softw. (Mar. 1992), 84-86.

[5] Ellis, C., Gibbs, S., and Rein, G. Groupware: Some issues and experiences. Communi. ACM, 34, 1(Jan. 1991), 39-58.

[6] Floyd, C., et al. Out of Scandinavia: Alternative approaches to software design and system development. Hum. Comput. Interaction 4(1989), 253-350.

[7] Gray, W.D., John, B.E., and Atwood, M.E. The precis of Project Ernestine or an overview of a validation of GOMS., In Proceedings of CHI'92. ACM, New York, 1992, 307-312.

[8] Greenbaum, J. and Kyng, M. Design at Work: Cooperative Design of Computer Systems. Lawrence Erlbaum Associates, Hillsdale, N.J., 1991.

[9] Greif, I. Computer-supported cooperative work: A book of readings. Morgan Kaufmann, San Mateo, Calif., 1988.

[10] Hefley, W.E. and Murray, D. Intelligent user interfaces. In Proceedings of 1993 ACM/AAAI International Workshop on Intelligent User Interfaces. ACM, New York, 1993.

[11] Helander, M. Ed. Handbook of Human-Computer Interaction. North-Holland, Amsterdam, 1988.

[12] Hollan, J., et al. An introduction to HITS: Human Interface Tool Suite. In Intelligent Interfaces. ACM Press, New York, 1991.

[13] Humphrey, W.S. Managing the Software Process. Addison-Wesley, Reading Mass., 1989.

[14] Kitson, D.H. and Masters, S.M. An analysis of SEI software process assessment results: 1987-1991. Proceedings of the 15th International Conference on Software Engineering. IEEE Computer Society Press, Washington, D.C., 1993, 68-77.

[15] Leveson, N.G. and Turner, C.S. An investigation of the Therac-25 accidents. IEEE Comput., 26, 7(1993), 18-41.

[16] Maybury, M., Ed. Intelligent Multimedia Interfaces. AAAI Press, Cambridge, Mass., 1993.

[17] McLaughlin, L.L. Multiple cooperating views: A new perspective for systems engineering. In Proceedings of the IEEE International Conference on Systems Engineering. IEEE, New York, 1989, 191-195.

[18] Myers, B.A. and Rosson, M.B. Survey on user interface programming. In Proceedings of CHI'92. ACM, New York, 1992, 195-202.

[19] Nickerson, R.S. Looking Ahead: Human Factors Challenges in a Changing World. Lawrence Erlbaum Associates, Hillsdale, N.J., 1992.

[20] Nielsen, J. The usability engineering life cycle. IEEE Comput. 25 3(1992), 12-22.

[21] Norman, D.A. and Draper, S.W. Eds. User Centered System Design: New Perspectives on Human-Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, N.J., 1986.

[22] Plaisant, C. and Shneiderman, B. Scheduling ON-OFF home control devices. In Proceedings of CHI'91 ACM, New York, 1991, 459-460.

[23] Platt, L. Presentation at Software Engineering Symposium, Software Engineering Institute, Pittsburgh, Penn., 1993.