information needed for design in a more time- and cost-effective manner that meets the constraints of fast-paced design and development projects. The particular methodology that should be used at any one time depends on the specific question to be answered and, often, on the time available to obtain the answer. Discussions of the appropriate match between design question and behavioral methodology are presented in Anderson and Olson (1985), Karat (1988), Landauer (1988), and Meister (1985).

Two categories of behavioral methodologies are especially useful for the human factors specialist involved in human–computer system design. One is a set of analytical methodologies that can be referred to generically as *task analysis*. These analytical tools enable the designer to understand the user's activities while performing a task, from the user's perspective. Task analysis is described in Section 6.1. The second category includes methods for testing and evaluating a system to determine if it is usable. These methods are described in Section 6.3.

4. Models of the Software Development Process

Most of the human factors activities described in this chapter take place in the context of the software development process. To be most effective, these activities must be appropriately coordinated with the activities of other project team members and be formally integrated into the project plan for the overall system. How that coordination occurs depends on the software development model that is being followed. A brief review of software development models is offered below. This review illustrates that models of software development have been changing in recent years in ways that better accommodate the type of design process that is required for interactive human–computer systems. Not coincidentally, the same time frame has seen the creation of design tools, i.e., user interface prototyping tools, that make the new models both possible and more effective; these will be discussed later in this chapter.

Software development models are used to structure the design and development process; they specify the order of the stages involved in software development and the transition criteria for moving from one stage to the next (Boehm, 1988). The model that is perhaps the most widely used is the linear or "waterfall" model. This model assumes that software is developed in successive stages, beginning with system planning and then proceeding through various stages of requirements, high-level design, detailed design, coding, integration, deployment, and operations and maintenance. (See Boehm [1988] for an overview of this model and others.) Although there may be feedback from one stage to the preceding stage, in general it is assumed that work will proceed sequentially from one stage to the next. This type of model is characterized as *top-down*: the system is specified at a general level first and specifications become increasingly detailed; when the design is sufficiently precise, it is implemented in code.

This type of sequential, noniterative design process has been successful for *noninteractive* systems, but has been less successful with *interactive* human–computer systems (Boehm, 1988; Grudin, 1991; Hartson and Hix, 1989). A primary problem with the waterfall model is its requirement that detailed documents be completed (e.g., requirements and design documents) before proceeding to the next stage. These documents are typically text-based documents that describe detailed components of the system, including the user interface. However, it is difficult to write adequate detailed requirements for user interfaces for interactive systems without first developing prototypes of the user interfaces. A user interface designer needs to be able to *see* the design, to work through user scenarios step-by-step, and to look across various user scenarios to ensure that the design is coherent and appropriately consistent. In addition, the user interface designer needs to collect feedback on the design from users to determine if it meets their needs and is usable from their perspective. Text-based requirements can lead to the generation of a large quantity of code underlying a user interface that is found—*after* development—to be difficult to understand and to use. The consequence at that point is that either: an inadequate product is delivered, or it takes longer to deliver the product because code must be modified to meet users' needs.

In addition, studies of the process of designing interactive computer software have revealed that a waterfall model is not always followed—even if it is stated as the appropriate design process (e.g., Hannigan and Herring, 1987; Hartson and Hix, 1989; Johnson and Johnson, 1989). Developers have reported that there is no uniform way of designing software; the process differs from designer to designer, stages in the process are not always discrete and do not always occur in the specified order, and there are iterations between some of the stages, depending on the product, design team, and schedule. Coding is often begun before requirements have been completed, or, as Hannigan and Herring (1987) put it, specifications are "refined, updated, reviewed, changed, disobeyed, etc." as development proceeds.

Models that are more iterative in nature and that involve all members of the design and development team working in parallel hold promise for delivering more usable human–computer systems, as well as for reducing development time and better supporting the actual work habits of designers and developers (e.g., Boehm, 1988; Hartson and Hix, 1989; Winner *et al.*, 1988).

Winner *et al.* (1988) describe a general approach to development called *concurrent engineering* that addresses some of the problems of the waterfall model. Concurrent engineering is a systematic approach to the integrated design of products and their associated processes, including manufacture and support. Key components of concurrent engineering include:

1. Consideration, from the initial stage of planning, of all elements in the system's life cycle from conception through disposal;
2. Multidisciplinary teams that identify all issues in a timely manner and that evaluate the impact and risks of various design alternatives;
3. An early and continuously increasing understanding of user needs throughout the process; and
4. Ongoing dialogue regarding the trade-offs among user needs, cost, schedule, and quality.

The concurrent engineering model assumes that there will be iteration of the design throughout the system development process, with increasing closure as the understanding of all relevant parameters increases. For example, user interface requirements might initially be high-level and general; as communication among user interface designer, project team members, and potential users continues throughout the design process, the requirements will become more specific and detailed.

Boehm's (1988) *spiral model* of software development is consistent with the concurrent engineering approach. The spiral model includes prototyping, testing, and iteration as key concepts. There are separate cycles for each phase of product design and development. Boehm's phases are investigation of the concept, requirements, design, and detailed design. Within each cycle, the same sequence of steps is addressed. The steps involve:

1. Determining the objectives, alternatives, and constraints for the portion of the system being addressed (e.g., performance, functionality, user interface);
2. Evaluating the alternative approaches and identifying and resolving risks;
3. Doing the appropriate type of development and testing for this phase (e.g., developing and validating software requirements); and
4. Planning for the next phase.

For example, if designing a poor user interface is a risk, Boehm (1988) suggests the project team might use the "risk management techniques" of prototyping, task analysis, user scenarios, and user characterization to ensure that a usable user interface is designed. In each successive cycle, the prototype becomes more detailed, and different specific risks might be identified, which could be resolved through user testing.

Hartson and Hix (1989) offer a model specifically for human–computer interface design that deviates even further from a sequential model. Their "star life cycle" places *evaluation* at the center of the other activities involved in user interface development. The model is called a star because evaluation is placed at the center of the star, with each of the other activities representing the points of the star. The other activities are:

(a) task and functional analysis,
(b) requirements and specifications,
(c) conceptual design and formal design representation,
(d) prototyping, and
(e) implementation.

According to the star model, these activities may occur in almost any order and alternation among them may be rapid. However, evaluation occurs between any move from one activity to another. For example, after modifying the prototype, evaluation would be conducted before detailing requirements based on the prototype. The evaluations might be minimal and informal (e.g., a peer review if only minor changes have been made to a prototype) or major and formal (e.g., usability testing with potential users if major changes have been made to a prototype).

Concurrent engineering, the spiral model, and the star model vary in their scope, with concurrent engineering pertaining to any type of engineering, the spiral model pertaining to software development, and the star model pertaining specifically to human–computer interface development. However, the models are consistent in emphasizing the importance of clarifying users' needs through iterative design and testing. Also, the star model of human–computer interface design is easily embedded within the spiral model, such that iterative prototyping and testing of the user interface occurs within any one general cycle of the spiral model; for example, the user interface prototype may be modified several times during initial planning, several times during the higher-level requirements phase, and several times during the design phases.

The growing awareness of the concurrent engineering and spiral models changes the design and development team's expectations about the value of iteration, making it easier for human factors specialists to integrate iterative prototyping and testing into the overall system development process, regardless of the specific model followed for a design and development project. Even within the outline of a waterfall model, rapid prototyping tools make it possible to design and test user interface prototypes iteratively during the requirements phase, i.e., during the same period of time that systems engineers and other team members are creating detailed functional requirements. The resulting user interface prototype may then become part of the package of detailed system requirements that are given to the software developers who implement the requirements in code. (For this model to be successful, it is imperative that the human factors specialist/user interface designer communicate frequently and well with the software developers to ensure that the prototyped user interface can actually be implemented under the various development constraints.) Within the outline of a concurrent engineering and/or spiral model, the iterative user interface design process tends to be somewhat easier, because the need for iteration and clear, continuing communication is better understood by all project team members, who may also be engaging in iterative design of their own components of the system.

5. Human Factors Activities in Human–Computer System Design

General principles for designing usable human–computer systems, described in Section 3, are carried out within the software development process described in the last section (Section 4). As members of design and development teams, human factors specialists have four major roles to play; these are described in Section 5.1. Human factors specialists complete these roles by performing the activities described in Section 5.2. These activities cover the entire design and development process, from planning to deployment. If all or most of these activities are performed, the probability of providing usable systems that adequately meet users' needs is greatly increased.

5.1 Roles of the Human Factors Specialist on Software Development Teams

The human factors specialist should play four primary roles, all of which are interrelated, on human–computer system design teams. These roles are:

1. Design of the human–computer interface;
2. Tester and evaluator of the user interface;
3. User advocate; and
4. Integral member of the design team.

Because design and evaluation should occur iteratively throughout the design process, these two roles are discussed under the same heading.

5.1.1 Designer and Evaluator of the User Interface

One primary responsibility of the human factors specialist should, ideally, be the design of the user interface (Chapanis, 1991). The human factors specialist is often the only member of the design team who brings a knowledge of human capabilities and human–computer interaction, an ability to locate additional information rapidly, and the skills in behavioral methodologies necessary to collect information from users during the design process. This behavioral information—combined with information from other team members—is essential for creating a user interface that is easy for people to learn and to use.

The claim that the human factors specialist should have a primary design responsibility is not meant to imply that the human factors specialist is the *only* team member with a role to play in design. Graphical designers may contribute significantly to the aesthetic appeal of graphical and multimedia user interfaces; software engineers and other engineers contribute information about software and hardware alternatives; technical writers may be responsible for creating instructional materials for use, installation, and maintenance; and training experts may develop training courses.

The design role involves both design and testing (or evaluation), conducted iteratively throughout the design process. Testing during design is sometimes referred to as *formative evaluation*, while testing of a completed design is called *summative evaluation*. The term *usability testing* is used in this chapter to refer to both formative and summative evaluation. Ideally, an initial design of a user interface is created, perhaps with a rapid prototyping tool; this design is then demonstrated to potential users and to project team members, whose feedback is used to modify the design. This process is repeated until the user interface design clearly meets users' needs. The user interface prototype then becomes a major segment of the user interface requirements for the system or, depending on the prototyping tool, is actually incorporated into the system. The process of iterative design and evaluation is widely regarded as critical for good user interface design, and will be emphasized throughout this article. (In Section 6.2, rapid prototyping of the user interface is described in more detail.)

As noted previously, the user interface includes not only computer hardware and software, but also instructional and technical support materials. Although the human factors specialist may assist in the design of instructional materials, the most typical human factors role is evaluation of the instructional and support materials to ensure their usability. Technical writers are often members of the project team, and are responsible for creating the instructional and support materials.

The role of user interface designer has only recently become common for human factors specialists. The roles most frequently played in the past were those of consultant and tester/evaluator. As a consultant, the human factors specialist provides information about human capabilities and user interface design issues to system developers, who have responsibility for both designing and implementing the user interface. The information provided by the human factors specialist may come both from the literature and from application of the behavioral methodologies. As a tester and evaluator, the human factors specialist tests and evaluates the user interface during or after its design. This is a model that has been prevalent in the design of large human–machine systems for the military (Meister, 1987). Unfortunately, human factors specialists operating as consultants and evaluators often find it difficult to influence design sufficiently to ensure usability, regardless of the size of the system being designed (Grudin and Poltrock, 1989; Meister, 1987; Meister and Farr, 1967). They are frequently brought in too late for both advice and evaluation. After decisions that severely constrain the user interface have already been made about the hardware and software platforms and after a significant portion of the system has already been coded, developers often find it impossible, given schedule constraints, to implement changes that are necessary to ensure that the system is usable.

The increasing availability of rapid prototyping tools for user interfaces has greatly enhanced the human factors specialist's ability to function as user interface designer. Such tools allow the user interface designer to design and test the user interface iteratively, and to specify (precisely) the look and feel of the user interface before implementation begins. In addition, some user interface prototyping tools generate code, and therefore the prototype can actually become the user interface of the final software.

5.1.2 User Advocate

Another primary function of the human factors specialist is that of *user advocate*. The human factors specialist must ensure that the needs of users of the system are given priority throughout its design. As a user advocate, the human factors specialist functions as a champion of "user-centered system design," i.e., design that is driven by the user's needs and that always considers the user's perspective (Norman and Draper, 1986).

To ensure that users' needs are met, the human factors specialist must understand the users (their basic perceptual and cognitive capabilities and their skills that are relevant to the user interface and to the particular tasks), the tasks they will perform with the computer system, and the environments in which the computer system will be used to perform the tasks (Bennett, 1984; Shackel, 1984, 1988). The human factors specialist must ensure that, throughout the design process, users' needs and concerns are given priority; all the other issues that vie for attention in system design (e.g., cost, schedule, performance) should be considered within the context of users' needs for usability as well as functionality.

5.1.3 Integral Team Member

The human factors specialist should function as a full team member on the project team, and should be involved throughout the entire system design and development process, from the planning stages through use of the system by its intended users. (If the project is large, there may be multiple human factors specialists, all of whom should function as full team members.) The design and development of a human–computer system is a multidisciplinary activity that requires specialists with many different skills (Catterall *et al.*, 1990; Fissel and Cecala, 1988; Kim, 1990; Laurel, 1990). A typical project team may include representatives from many organizations, including marketing, systems engineering, human factors, industrial design, graphic design, technical writing, training, development, and operations and maintenance. These representatives bring different areas of complementary expertise to the team. It is important that the relevant expertise be available when it is needed. The expertise of the human factors specialist (as well as that of many other team members) is needed throughout the process.

There are two major reasons the human factors specialist needs to be involved throughout the process. First, many design decisions affect the user interface, and information about the impact of these decisions on the interface must be provided in a timely fashion. Second, information from users should be collected throughout the design process—initially to identify their characteristics, skills, needs and tasks and later to assess the extent to which the evolving design meets their needs and is usable.

Timely Information. The user interface is affected by many team decisions made throughout the design and development process. Often the human factors specialist is the team member who is most knowledgeable about the effect of software and hardware decisions on the user interface; therefore, the human factors specialist should be involved during such decision-making to identify the decisions that will affect the user interface and contribute to the decision-making process. Figure 3 illustrates some of the information and constraints that influence the user interface. The human factors specialist must be able to provide the following types of information at the right time to affect system design:

- Information about users that supplements information provided by a marketing organization, such as information about users' tasks, users' task-relevant skills, and users' environments. This user information

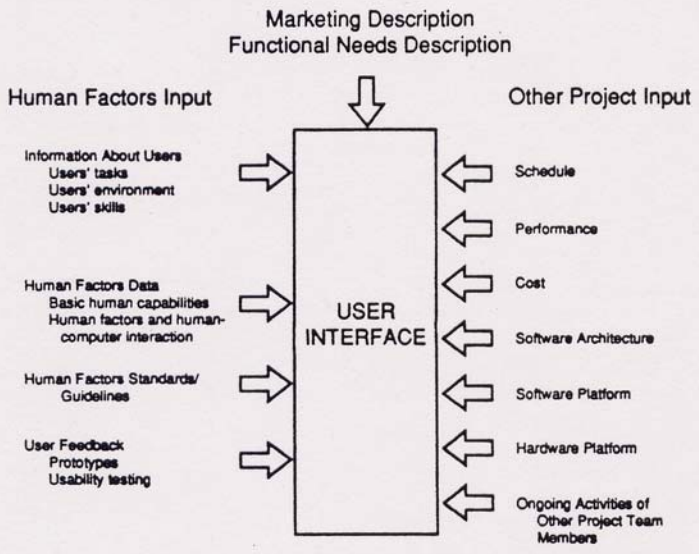

FIG. 3. Information and constraints that shape the user interface.

may influence decisions about system functionality and hardware and software platforms, in addition to the design of the user interface.

- Analyses of the impact to the user interface of potential decisions about other components of the system (such as software platform, hardware platform, and software architecture) and about components of the project plan (such as schedule, development costs, and overall system costs).
- Analyses of the advantages and disadvantages of various approaches to implementing the user interface, e.g., character-based versus graphical user interfaces.

Information from Users. Second, feedback from users should be collected throughout the design process. Different information is needed from users at different phases in the design process. For example, information about users' tasks and environments is required early in the process, but data on users' responses to user interface prototypes should be obtained throughout design and development. Different behavioral methodologies are required depending on the type of information needed at each phase in the development process. The human factors specialist who fully participates in the project team will be present to provide essential information about users and the user interface at the right time.

A common problem for human factors specialists—and for the projects on which they work—is that they become involved in the development process too late. Grudin and Poltrock (1989) conducted a survey in seven large corporations to investigate the roles and activities of different professionals involved in user interface design. They found that 100% of the human factors respondents wanted to be involved in projects before implementation (i.e., coding) of the user interface was begun, but only 57% reported being involved in projects this early.[2] Furthermore, only 27% of the human factors specialists reported that their activities were "always or usually" successful when their involvement began after implementation was complete, i.e., when they were brought in to evaluate the final product. Twenty percent said they were "occasionally" successful, and 53% said they were "rarely or never" successful. The software engineers were even more negative in their evaluations of the effect of the late involvement of human factors specialists. Only 10% of the software engineers said human factors specialists were "always or usually" successful if involved after implementation had begun; 25% said they were "occasionally" successful, and 66% said they were "rarely or never" successful. The evaluations of the respondents from marketing were similar to those of the evaluations from software engineers. However, respondents reported more successes when involvement began earlier in the development project. When asked how often the projects had successful outcomes when human factors activities began in the middle of development projects, 43% of the human factors specialists responded "always or usually," 40% responded "occasionally," and 17% reported "rarely or never." It might be expected (and has been the experience of many human factors specialists) that involvement at the *beginning* of a project is even more successful. The Grudin and Poltrock (1989) data confirm that involvement should begin early and be continuous to have maximum impact on the user interface and on usability. Early involvement requires that marketing, systems engineering, and development, as well as human factors specialists, recognize its importance.

5.2 Human Factors Activities at Each Phase of Development

Carrying out the roles just described has different implications at each phase of the development process, when different tasks may be completed or different methodologies used. Five general phases of system design are shared by virtually all software development process models, although the

[2] Technical writers, who are responsible for user documentation, a component of the user interface, would also benefit from being included earlier in the development process; 87% wanted to be involved before implementation, but only 28% actually were.

extent to which these phases occur in parallel or overlap varies among models. These phases are: planning, design, implementation, testing, and deployment. As is highlighted in Boehm's (1988) spiral model, each of the first four phases may occur iteratively, but at a greater level of detail, as system design and development progress. At each of these phases, there are important activities for the human factors specialist to perform in collecting information from users and in design of the user interface. The major activities are listed in Table II. As this table highlights, the same general type of human factors activity (e.g., user needs analysis, user interface prototyping) may occur during several of the phases, although the *particular* methodology or the way it is employed will probably vary depending on many of the specific characteristics of the system and project (e.g., new system or later release, phase of system development, project schedule, homogeneity or heterogeneity of user population, prior knowledge of users, old or new technology, etc.). Because of the variety of system- and project-related factors that influence selection and use of methodology, it is impossible to provide one "recipe" for which specific methodology should be used when and in what manner. This is why skill and training in human factors is critical; one cannot simply "follow the rules" because every new development project requires some wisely considered exceptions to the rules. Thus, the human factors specialist should be familiar with the wide variety of methodologies and their use, so that the right one can be used at the appropriate time in the most useful manner. Each of the major human factors activities listed in Table II is described in more detail in Section 6 of this chapter.

The human factors specialist, as a primary user advocate, must ensure that data about the users and their needs are available as required throughout the process and must ensure that these data are appropriately translated into design. To be effective, the human factors specialist must not only consider the users' needs, but must also consider the trade-offs among user needs and

TABLE II

HUMAN FACTORS ACTIVITIES DURING SYSTEM DESIGN AND DEVELOPMENT

Phases of system development	Human factors activities		
	User needs analysis/ task analysis	User interface design/prototyping	Usability testing
Planning	×	×	
Design	×	×	×
Implementation		×	×
Testing			×
Deployment	×		×

other project goals and constraints (such as performance, cost, and schedule). Thus, the human factors specialist must look both *toward the user* and *toward the project team* throughout the system development process. Below is a high-level (and unusually complete, relative to "real-world" practice) description of the human factors specialist's activities at each phase of the system development process.

5.2.1 Planning

During the planning phase, the human factors specialist's primary activities are:

- To collect information about users' characteristics, tasks, environment, and needs;
- To incorporate this information into potential user models and high-level user scenarios; and
- To create preliminary user interface prototypes.

Table III outlines these activities, along with some of the behavioral methodologies that may be employed during this phase.

TABLE III

PLANNING PHASE—UNDERSTANDING USER NEEDS

Human factors activities	
Looking toward the user	Looking toward the project team
• Collect information about users—their skills, tasks, and environment • Analyze user interfaces of competitive systems • Create preliminary user models • Create high-level user scenarios • Create preliminary user interface prototypes • Demonstrate the user interface prototypes to users	• Understand project goals (functionality, schedule, costs, etc.) • Include human factors activities in the project plan • Inform the project team of implications for users and for the user interface of hardware and software options • Demonstrate the prototype to project team members
Behavioral information and methodologies	
Literature searches (human capabilities and limitations, human–computer interaction), task analysis, competitive usability analysis, user profiles, interviews, focus groups, naturalistic observations, rapid prototyping	

While collecting information on user needs, the human factors specialist obtains information about the users' skills, the tasks they currently perform with their existing system and will perform with the new system, and their environment (other computer systems in use, work processes and procedures, etc.). This information can be used to create descriptions of the users' conceptions of the system and their tasks with the system (*user models*), descriptions of the users' tasks in relation to their capabilities (*task analyses*), and preliminary step-by-step descriptions of the way the new system might be used to perform tasks (*user scenarios*). With this information, along with preliminary information from the project team, the human factors specialist may create preliminary user interface prototypes. These prototypes may be shown to users to collect additional feedback.

Another activity that is sometimes performed by human factors specialists is analysis of the user interface of competitive systems. In some organizations, competitive analyses are performed by market researchers or by human factors specialists and market researchers working collaboratively. An analysis of the strengths and weaknesses of existing, competitive designs can be an excellent source of good ideas and can help to prevent serious design problems. Usability testing (described in Section 6.3) can be performed with competitive systems to help specify usability goals for the system being planned.

While obtaining information about users and competitive systems, the human factors specialist maintains close contact with the project team. He or she must understand the project goals and must ensure that human factors activities are included in the project plan. As various hardware and software options are discussed (such as the choice of hardware and software platforms), the human factors specialist seeks to understand their implications for users and the user interface, and informs the project team of the implications. The information obtained about the users is shared with the other team members to maintain a continuing focus on the user. The user interface prototypes are demonstrated to team members to improve communication and to ensure that there is a common vision of the system being designed.

5.2.2 Design

During the design phase, the human factors specialist continues most of the activities begun during the planning phase, but at a more detailed level (see Table IV). He or she continues to collect information on users' needs. This may involve collecting more specific information from users about the

TABLE IV

DESIGN PHASE—ITERATIVE PROTOTYPING AND USER FEEDBACK

Human factors activities	
Looking toward the user	Looking toward the project team
• Develop user models and user scenarios in more detail	• Demonstrate prototype to project team members
• Set usability objectives	• Understand project design constraints and provide information on their implications for user interface design
• Specify usability test plans	• Participate in determining trade-offs and in design problem solving
• Refer to and/or create user interface standards and guidelines	• Create user interface requirements
• Create detailed user interface prototypes	• Hold walkthroughs with project team members
• Collect feedback on the prototype from users	• Add user scenarios to the system test plan
• Engage in iterative design and testing	• Conduct expert reviews by other user interface designers
• Conduct laboratory experiments where necessary	• Coordinate work on all aspects of the user interface
Behavioral information and methodologies	
Literature searches (human capabilities and limitations, human–computer interaction), task analysis, competitive usability analysis, user needs analysis, user interface standards and guidelines, demonstrations, usability testing, experimental tests, questionnaires, interviews, observations, thinking-aloud techniques, verbal reports	

implications of design issues under consideration, and it may involve again consulting the literature on human capabilities and human–computer interaction. More detailed user models and user scenarios are developed to support the design of more detailed prototypes. User interface guidelines and standards are identified or created to support the design and development of a consistent and usable user interface. In addition, usability objectives for the system are specified, and work begins on a usability test plan that will become one component of the final system test plan.

At this phase, the iterative process of design and collection of user feedback is critical. As the user interface prototype is developed it is demonstrated to users, and the prototype is then revised based on the user feedback. If the prototype or components of the prototype are functional (i.e., if the prototype can actually be used), users are asked to perform common or critical tasks with the prototype. This may include initial testing to determine if usability objectives are being met. User performance data, such as task completion time and errors, are collected, along with users' opinions and preferences. Portions of the user instructional materials may also be tested.

Relevant external and company user interface standards are identified at the beginning of the design phase, and the user interface is designed to be consistent with the standards. Throughout the evolution of the design, compliance with the standards is assessed. If the standards are not sufficiently complete, it may be necessary to supplement them with additional standards that are specific to the product line. It may also be necessary to assess consistency with other systems that have been developed as part of the same "family" of systems or products, to ensure appropriately consistent user interfaces across systems.

Communication with other members of the project team remains critical. The evolving prototype is shown to marketing representatives on the team to ensure that the system being developed is what marketing had intended. The prototype is demonstrated to systems engineering, development, and system test members of the project team, both to facilitate communication and, if the prototype itself will *not* become part of the final system, to ensure that the design can actually be implemented. The human factors specialist works with other members of the project team on a continuing basis to understand hardware and software constraints, other project constraints, and to help in problem solving. Often the human factors specialist works with other project team members in addressing problems that cross the boundaries of functional specialties, as many problems do.

A primary deliverable of this phase is often a set of human–computer interface requirements that specify precisely how the human–computer interface of the final system will "look and feel" (i.e., the appearance and the user–system dialogue of the human–computer interface). The requirements may incorporate the prototype along with additional textual or graphical information about user–system dialogue or other information that cannot easily be conveyed explicitly with the prototype. If the rapid prototyping tool used is one that generates code that can be used in the final system, the prototype may be handed off to system developers to become integrated with other system components.

As the prototype, which will become part of the human–computer interface requirements, is being developed, reviews of the prototype are held with team members. If a human factors community exists, reviews or "walkthroughs" of the user interface are held with other human factors experts (e.g., Jeffries *et al.*, 1991). In addition, there should be at least one formal review or walkthrough with other members of the project team, especially the system developers, to ensure that the requirements are understood by all and that no major implementation problems are expected.

During this phase, if not before, plans are made for designing all other aspects of the user interface, such as user instructions, training materials, and maintenance instructions. Work on all components of the user interface is coordinated to ensure consistency from the user's perspective.

5.2.3 *Implementation and Testing*

While the system is being implemented and tested, the human factors specialist continues to work with both users and team members (see Table V). The iterative design process should not be considered complete; the user interface prototype or evolving final system should continue to be tested with users whenever reasonable. The cost of fixing problems identified early is much lower than the cost of fixing problems after development is complete (e.g., Mantei and Teorey, 1988). The human factors specialist works with developers to resolve the problems that inevitably surface during development. As a part of system test, usability testing is conducted to ensure that the final system meets the usability objectives that were set in the design phase. Usability testing should include tests of instructional materials as well as the human–computer interface.

5.2.4 *Deployment*

When the system is introduced to the target users in "alpha" and/or "beta" tests (i.e., limited introductions before the system is made generally available), it is especially important that human factors specialists be

TABLE V

IMPLEMENTATION AND TESTING PHASES—ITERATIVE DESIGN AND USER TESTING

Human factors activities	
Looking toward the user	Looking toward the project team
• Continue iterative design and testing as needed • Conduct usability testing of all components of the user interface	• Work with software developers to resolve problems • Participate in system test of the user interface, working through the user scenarios • Ensure all components to the interface are consistent
Behavioral information and methodologies	
Usability testing, questionnaires, interviews, observations, verbal reports	

TABLE VI

DEPLOYMENT PHASE—USER FEEDBACK

Human factors activities	
Looking toward the user	Looking toward the project team
• Conduct usability testing at the users' locations • Observe users performing actual work with the system	• Provide results of user testing to project team members • Help resolve any major problems identified at this phase • Summarize data for use in next product or next release
Behavioral information and methodologies	
Usability testing, questionnaires, interviews, observations, verbal reports, task analysis	

involved in the evaluations (see Table VI). No matter how exhaustive the iterative design process has been, unexpected problems occur when users begin to rely on the system to perform their tasks. Methods of collecting data may include observations of people using the system to perform their daily tasks, performance on a series of tasks selected to uncover hidden problems, and users' responses to interviews or questionnaires. If problems are identified at this stage that will significantly decrease usability, changes will be required before general deployment. (If the process of iterative design and testing has been followed, such unpleasant surprises should not be numerous, although some will undoubtedly occur.)

Even after the system has been in use for some time, additional data on the system's use and its usability are collected. After a system has been used extensively, users often identify additional or different problems than they noted when they first began to use the system. This information can be fed into the design process for new releases of the system or for new systems.

5.3 Idealized, but Meeting a Real Need

The preceding description of human factors involvement in the system design and development process is idealized. This description captures what *should* happen rather than what typically *does* happen. Human factors specialists are often *not* integrally involved from system planning to deployment, just as few development projects proceed as they ideally should (Grudin, 1991; Grudin and Poltrock, 1989). Furthermore, even when human factors specialists are integral team members, as is increasingly the case in many organizations, time is frequently too short to incorporate all the user-centered design activities cited above. Iterative design processes, that would prevent major surprises for the project team and frustration for users, are still not commonplace. And there is often inadequate recognition by management of the criticality of these processes.

However, on the positive side, it is increasingly being recognized that human factors involvement is valuable in ensuring a *usable* system, and users are clamoring for usable systems. In addition, as noted previously, new models of the software development process reflect the need for more attention to user requirements and for iterative design. In many companies, human factors specialists and other champions of user-centered design are successfully incorporating more and more components of the scenarios previously described into their design and development processes (Bias and Alford, 1989; Fissel and Cecala, 1988; Flamm, 1989; Hawkins, 1989; Rideout *et al.*, 1989; Riley and McConkie, 1989; Vorchheimer, 1989; Whiteside *et al.*, 1988).

6. Human Factors Methodologies for Human–Computer System Design

In this section, a summary is given of several key methodologies and approaches used by human factors specialists during the software development process: task analysis, rapid prototyping of the user interface, and usability testing. These are not the only methods used in system development environments, but they are key methodologies used by the human factors specialist. Task analysis and usability testing are analysis and evaluation methodologies, respectively, that have their foundations in the behavioral sciences. Rapid prototyping involves the use of rapid prototyping tools to create early designs of user interfaces that can be tested with users and then modified. As mentioned in the previous section, many of these methodologies would be used more than once, in an iterative fashion, in an ideal design environment.

6.1 Task Analysis

One key design principle espoused by Gould and Lewis (1985) and others is an early and continual focus on users and their tasks. *Task analysis* is one method that has been used successfully to maintain this focus.

6.1.1 Definition and History

Task analysis refers to a class of methodologies used to understand the human component in a human–machine system. The basic goal of task