List of Suggested Reviewers

The following individuals are highly qualified to review this proposal:

Professor Alan Borning
Department of Computer Science
FR-35
University of Washington
Seattle, WA 98915
Phone: 206-543-6678; email: borning @ cs.washington.edu

Dr. Ruven Brooks
Schlumberger Austin Research
8311 RR 630
Austin, Texas 78726
Phone: (512) 331-3729; email: brooks@Austin.sar.slb.com

Dr. Bill Curtis
TeraQuest Metrics & SEI-CMU
3644 Ranch Creek
Austin, Texas 78730-3701
phone: 512-346-2435; email: curtis@acm.org

Professor Gerhard Fischer
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309-0430
Phone: (303) 492-1502; email: gerhard@cs.colorado.edu

Professor Jim Foley
GVU Center, College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
Phone: 404-853-0671; email: foley @ cc.gatech.edu

Professor John Pugh
School of Computer Science
Carleton University
Ottawa, Canada K0A 2H0
Phone: 613-788-4330; email: john_pugh @ carleton.ca

Professor Ralph E. Johnson
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
email: johnson@cs.uiuc.edu

A. PROJECT SUMMARY

Interactive computing systems are only as good as the tasks that they support. Designers of interactive systems have recognized this and begun to incorporate *use-oriented* design representations into their system development activities. One such representation is a task *scenario*, a narrative describing one or more users interacting with a computer to accomplish some task. But despite the increasing importance of scenarios to design practice—they can be used to express requirements, to envision new designs, to communicate design ideas to users, to evaluate prototype systems, to test theoretical models—there is no systematic, integrative methodology or framework to guide this practice. Scenario generation and application remains an art.

This research seeks to develop such a framework. It uses converging methods to systematize the concept of task scenario, to articulate the properties that make a scenario good for one design activity vs. another, and to develop tools and techniques that will guide designers in effective and efficient application of these use-oriented design representations. Studies of scenarios created in representative design projects, combined with observational studies of the process of scenario generation and comprehension, lead to an understanding of the key dimensions influencing scenario effectiveness. Scenario-based participatory design sessions provide an opportunity to test the impact of these dimensions in a real-world design setting. Tools that offer scenarios as an integrative representation for the entire software development life cycle help to increase the impact of use-oriented concerns on system design.

A critical component of developing any new design method is conveying the method to designers, to the individuals who will be designing the interactive systems of the future. Currently, students of software development are taught a process in which usage requirements are assessed in one step, and software is designed to meet these needs in another step. Scenario-based methods can be used to better integrate these two steps, and in so doing to provide for a mutual exchange of constraints and opportunities between the task and software domains. This education plan takes the fruits of the research on scenario-based design methods and feeds them into the development of educational materials for university students and software design professionals.

B. TABLE OF CONTENTS

Career Development Plan

1.0 Research Plan **C-1**

1.1 Objectives and Significance of the Proposed Research **C-1**

1.2 Relationship to Current State of Knowledge **C-2**

1.3 Plan of Work **C-5**

1.4 Prior Research Accomplishments **C-9**

2.0 Education Plan

2.1 Objectives of Education Activities **C-10**

2.2 Planned Education Activities **C-11**

2.3 Planned Teaching Activities **C-13**

2.4 Summary of Education Accomplishments **C-13**

3.0 Departmental Endorsement **C-15**

Bibliography **D-1**

Biographic Sketches of Senior Personnel

M. B. Rosson **E-1**

J. M. Carroll **E-3**

Budget & Budget Justification **F-1**

Current and Pending Support **G-1**

Facilities, Equipment and Other Resources **H-1**

Special Information Supplementary Documentation **I-1**

C. CAREER DEVELOPMENT PLAN

1.0 RESEARCH PLAN

The research proposed here represents an effort to recast and extend our research on scenario-based design (which was initiated in the industrial research setting of the IBM T. J. Watson Research Center) in a way that exploits the challenges and opportunities of the university setting and community provided by Virginia Tech.

1.1 Objectives and Significance of the Proposed Research

In contemporary design practice for interactive systems, task scenarios — narratives describing one or more persons interacting with a computer in the course of carrying out a task — are a medium for reasoning about user activities from a variety of perspectives. They are used for requirements capture and analysis, communication between users and designers, system envisionment and development, documentation and training, usability evaluation. Scenarios have focused the efforts of many in human-computer interaction (HCI) toward an integrated design framework (Carroll, 1990).

Nevertheless, scenario-based design methods are not a mature technology; observers are often confused by the diverse uses of scenarios (Campbell, 1992). For example, there is little consensus about the special utility or limitations of different types of scenarios. Should scenarios be couched at the level of self-conscious human activity, the level of input and output transactions, or both? What implications arise from couching scenario descriptions at different levels? What are the procedures and representations for doing this? Should we be using different sorts of scenarios of at different points during system development? Despite the growing emphasis on the use of scenarios in design practice (see, e.g., the discussion in the April 1992 issue of *SIGCHI Bulletin*), there have been virtually no efforts to develop a systematic framework to guide the generation and application of these rich, use-based design representations.

One objective of the current project is to develop such a framework. Through field studies of actual design projects, complemented with more controlled laboratory studies of scenario generation and comprehension, we will articulate the ways in which scenarios contribute to system development, scenario characteristics that aid or interfere with such contributions; we also will develop a preliminary process model of scenario generation and comprehension.

Scenario-based approaches to design also raise a serious — indeed classic — system development issue: Will an increased reliance on scenario representations *increase* the difficulties in making the transition from use-oriented specifications and software-oriented implementations? While much work in software engineering in past decade has been directed at narrowing this gap (Harrison & Thimbleby, 1989; Jacob, 1983), it seems prima facie that the current movement toward rich, context-specific usage descriptions may widen it anew.

A second objective of the current project is to develop and evaluate tools for scenario-based design that will *narrow* the gap between use-oriented and software-oriented design views. In particular, we will examine the role of object-oriented scenario implementations as a vehicle for integrating task and software concerns.

1.2 Relationship to Current State of Knowledge

The design of useful and usable interactive systems is a tremendously difficult task. Part of this is due to the inherently difficult characteristics of design in general: design problems are never specified adequately, and neither the possible design moves nor the final goal can be known in advance (Reitman, 1965). In the design of interactive computing systems, though, these general problems are aggravated by issues associated with software design — the systems under design are normally complex and often in constant flux; the design process requires management of tradeoffs among a variety of concerns (e.g., cost, reliability, usability); the size of the problems typically demands contributions from many individuals with diverse backgrounds and skills; and the system that results can be expected to have wide-ranging and ongoing impacts on human activity.

Faced with the job of designing systems where the desired result is a complex and moving target, the developers of interactive systems have found that task scenarios help to make both the problem description and the envisioned solution more concrete and manageable (Rosson, Maass, & Kellogg, 1990). Scenarios are used to illustrate aspects of the system under design, as a sort of functional specification (Carroll & Rosson, 1990; diSessa, 1991). They are used to communicate the design to prospective users (Ehn, 1988), or to engage users in cooperative design (Greenbaum & Kyng, 1991; Kyng, 1991). Scenarios also support formative and summative evaluation in design (Carroll & Rosson, 1992; Roberts & Moran, 1983).

Scenario: Adding the Lange & Moher CHI'89 conference paper

I just came across Beth Lange's 1989 CHI paper again, and decide I want it in my database. I go over to my bibliography window and pull up a new conference paper template. I enter the information in the order I remember it, first the authors, then the paper title, conference and date. I can't remember the location, so I leave it blank for now. At this point, I notice the "nickname" field and realize I haven't yet given it a nickname. I stop for a minute and think, then decide to call it "copy-edit" to identify it as one of the first documenting this OOP strategy. I wonder for a minute if I have already given this nickname to something else, but when I try it, the system accepts it, so I figure it must be unique.

Figure 1: Sample "basic-level" scenario from a bibliography design project

However, while there are many examples of contributions made by scenarios in design, there is no systematic methodology or scientific foundation to guide this practice. In our prior work, we have suggested that in the most useful scenarios for interactive system design are those that describe user tasks at a "basic" level — the level at which users construe their activities to themselves (see Figure 1 for an example of such a scenario developed in the course of building a bibliography utility, (Rosson & Carroll, 1995)). We have appealed to the work of Rosch and her colleagues in making this proposal (e.g., Rosch, Mervis, Gray, Johnson, & Boyes-Braem, 1976), but have not yet operationalized the notion of "basic" as applied to user tasks for interactive computing systems. And although we have analyzed our own use of basic-level scenarios in various design projects addressing the learning and use of object-oriented programming environments (Carroll & Rosson, 1991; Carroll, Singley, & Rosson, 1991; Rosson & Carroll, 1993b), we have not studied the extent to which such scenarios are useful to other designers working in other problem domains. Finally, while we have developed considerable experience in generating such scenarios for purposes of design envisionment, and in creating such scenarios as a means of organizing empirical observations (Koenemann-Belliveau, Carroll, Rosson, & Singley, 1994), we have not analyzed the cognitive processes involved in the generation or empirical analysis of scenarios in the course of design. The development of an adequate scenario set remains something of an art.

The research proposed here represents a systematic effort to move beyond scenarios as an art, to develop a framework that could be used to guide the generation, analysis and application of these design representations in interactive systems development. We will approach this goal with converging methods. We will collect examples of scenarios that have been used is actual development projects and analyze them in search of general characteristics. We will also carry out scenario generation experiments in which expert designers are studied in the process of generating scenarios under various conditions, and in which users and designers cooperate in developing scenarios. The scenarios generated by designers and by groups of designers and users will be evaluated with respect to their ability to communicate an envisioned design, both to potential users and to software developers. The result will be a taxonomic analysis of scenarios (qualified as necessary by design role and problem domain), complemented by a preliminary process model of scenario generation and comprehension.

Understanding and systematizing the use of task scenarios in design represents an important step toward a software development process that is use-oriented. However, this in itself does not address the long-standing concern about how to effect the transition from usage requirements to software implementation (see, e.g., (Swartout & Balzer, 1982). In traditional software development, requirements are provided in the form of "functional specifications", an item-by-item listing of the features to be provided by the implemented system. Such a listing is not user-centered, but it is easily understood by developers charged with designing and creating the software — the specification lays out in a well-ordered non-redundant format exactly the features that are expected in the system. In contrast, task scenarios embed specific system features within an often complex narrative; the narratives are intended to illustrate how individual features affect a user's experience in the context of a specific task, and how the features interact within and across tasks. Prima facie the job of extracting and implementing a set of features embedded within and distributed across a set of scenarios seems considerably more demanding than that of implementing a list of system functions.

Our work addresses this potential widening of the specification-implementation gap by including object-oriented implementation activities as a key element in our scenario-based design approach. At the same time that researchers in HCI have been investigating the role of scenarios in designing human-computer interaction, researchers studying object-oriented design (OOD) have been developing scenario-based approaches to software design: scenarios are often used in developing a problem domain model, which is then further elaborated and refined to build an object-oriented design (Gibson, 1990; Jacobson, Chriserson, Jonsson, & Overgaard, 1992; Rubin & Goldberg, 1992; Wirfs-Brock, Wilkerson, & Wiener, 1990). By using scenarios as a source of important problem objects and their interactions, these approaches begin to narrow the gap between specification and implementation. However, the OOD methods are not focused on reasoning about user tasks — they use the scenarios to gain insight into useful *software* abstractions.

Recently, we have proposed a merging of the scenario-based design approaches being developed in HCI and in OOD (Rosson & Carroll, 1993b; Rosson & Carroll, 1995). Briefly, we have illustrated how software developers might use the *same set* of scenarios as a medium both for analyzing and envisioning user tasks, and for analyzing and building an object-oriented software model implementing these tasks. Development of tasks and software would proceed in parallel, allowing the design team to recognize and address the interdependencies between tasks and software early and continuingly in the design process.

The interdependencies between user tasks and the software designed to support them are well-known. Tasks set requirements for new systems; as systems are developed, their software and hardware characteristics create constraints or opportunities for the tasks (Carroll & Kellogg, 1989). Many development projects manage these interactions through tight feedback loops between task specification and implementation, codifying

specifications in prototypes early and repeatedly to "try out" on users (Myers & Rosson, 1992; Wasserman, 1982). The approach we propose can be seen as a refinement of rapid prototyping, where our focus is on the prototyping of *tasks*. We are seeking to support a process in which designers experiment early and often with software abstractions, but always in the context of specific task requirements. So, for example, when generating an object model for the scenario sketched above (see Figure 1), the designers would work directly from a usage description and analysis that reflects hypotheses about the *user's* concept of a reference (e.g., references contain components such as authors and title, and reference-entry consists of specifying these parts). This would motivate and rationalize a software design in which these components are easily identifiable and manipulable. This is a common sort of dependency between user tasks and the software supporting them, where some aspect of the user's task structure must be respected or reified in the software model.

We have investigated such an integration of task and software design in our work on the Scenario Browser, a design environment for Smalltalk applications (Rosson & Carroll, 1993b; Rosson & Carroll, 1995). In that environment, design begins with the specification of one or more task scenarios. As soon as a scenario has been described — even at a very course grain — the designer can begin to develop and refine a set of Smalltalk objects that could be used to implement the scenario. The environment provides facilities for the software developer to refine and evolve the object model of a scenario as the task described in the scenario becomes better specified; it also provides tools for recording the design rationale for both task and software features. In our demonstration project, we illustrated how the shared usage context provided by a scenario not only helped in the identification of important objects and their relationships to one another, but also allowed our software design activity to feed *back* into task design: an object modeled for a scenario may be given "responsibilities" that go well beyond the its referent object in the real world, suggesting new ways of conceptualizing user tasks (Rosson & Alpert, 1990; Taichritzis, Fiume., Gibbs, & Nierstrasz, 1987).

Our work with the Scenario Browser demonstrated how a single designer might use such an environment to integrate the specification and implementation of a relatively simple application (our example was a personal bibliography tool). However, this work also raised many questions concerning the integration of tasks and software in system development more generally. Can a single set of task scenarios really serve both purposes? Are some scenarios most useful for considering usage issues and others for software issues, and if so, how do we synthesize different scenario purposes? Software design has a strong requirement for completeness — how do we address this issue with respect to a necessarily incomplete set of usage episodes? How do we ensure consistency across a large number of scenarios, especially when design work is being carried out as a group effort? What contributions can the object-oriented paradigm make to task design? What sorts of methods or tools will most encourage software designers to consider the HCI issues inherent in a scenario, and vice versa? How can we best train designers in this integrative approach?

The tool development work in the current project will examine some of these issues. We plan to take the lessons we have learned in the Scenario Browser project, combine that with what we learn about scenario generation and application in the studies described above, and build a robust environment for the integrated development of tasks and object-oriented software. Because this scenario-based environment will reflect a novel approach to software development, it will be difficult to assess the value the tools for designers already experienced in more traditional development approaches. Thus our empirical evaluation of the design environment will focus on its effectiveness in *teaching* the simultaneous consideration of task and software design issues in interactive systems development.

1.3 Plan of Work

The planned work will require careful sequencing and integration over the three years of requested support. The early work will concentrate on characterizing the nature and generation of scenarios; the results of this work will be used to guide the participatory design project as well as the design of the scenario-based design tool. Later in the project, the focus will turn toward evaluation, both with respect to the effectiveness of scenarios as a design representation and with respect to the scenario-based design tools built as part of the project. Major milestones in the five areas of proposed work are graphed in Figure 2. We turn now to a more detailed account of each of these activities.

Research activity	**S95**	**F95**	**SP96**	**S96**	**F96**	**SP97**	**S97**	**F97**	**SP98**
Scenario survey									
initial usage survey	X								
sample scenario collection		X							
analysis			X						
Lab study - generation									
data collection		X							
data analysis			X						
Cooperative design with community									
tool creation, scenario creation	X	X							
consultation with Aarhus colleagues		X							
participatory sessions, prototypes			X	X					
data analysis				X	X				
Lab study - transformation									
data collection					X	X			
data analysis						X	X		
Scenario-based design tool									
preliminary design	X	X	X						
implementation		X	X	X	X				
evaluation						X	X		
redesign								X	X

Figure 2: Milestones for proposed research activities broken down by academic term

Scenario survey. This survey of practice will take place in two phases. First, practitioners from the HCI design community will be contacted via email and requested to fill out a short survey concerning their use of scenarios in different phases of system development. The survey will also request individuals to indicate if they possess scenarios that they would be willing to submit to a content analysis.

Based on the results of the first survey, we will collect sample scenarios, along with brief descriptions of the project context and outcome. Some participants will be recruited for more intensive follow-up phone interviews. We will carry out a multidimensional content analysis of the scenario materials, characterizing both the content and style of the scenarios themselves, and the extent to which scenario attributes vary as a function of design purpose, project variables and project outcome. (Note that as part of my current volunteer efforts as Papers Chair for CHI'95, I have assembled a database of over 400 HCI professionals, and through this have contact information for over 200 individuals working in companies or industrial research; I will use these individuals as the starting set for my survey requests).

Scenario generation. Pairs of individuals — a software designer and a problem domain expert — will be asked to work together to generate 5-8 user task scenarios based on an open-ended problem description. We will select several different problem domains (e.g., CAD, personal organization tools, educational software), so as to assess the extent to which domain-specific characteristics might influence scenario generation. The pair's

discussion will be recorded, as well as any design artifacts (e.g., sketches, lists) produced in the design session. Subsequent analysis of the design protocols and artifacts will be used to develop a preliminary process model of scenario generation, and to develop recommendations for how such a process might be supported or enhanced via new methods or tools. Given the richness of design protocols, and the resulting complexity of analysis, we plan to study only 6-8 designer pairs for each problem description.

Cooperative design with Blacksburg community members. Blacksburg is home to a major research and development project on community networks — the Blacksburg Electronic Village (BEV) — which has received considerable attention both locally and at the state and national levels. The network was put in place with the help of the local telephone company, and its operations are managed by a group within Virginia Tech's Information Systems department. It currently has over 9000 registered users and is expanding rapidly; a unique aspect of this particular community network project has been its ability to provide high-speed network connections (Ethernet) to a large number of community members.

The BEV has generated considerable excitement within a number of research constituencies at Virginia Tech (e.g., education, computer science, industrial engineering, anthropology, science and technology studies), and projects ranging from network-based courseware development, social implications of educational networks, and technology and policy evolution are underway. It is unusual to see such an intensive and broad-based spirit of cooperation between Computer Science researchers and the general public. The current project proposes to leverage the work on the infrastructure already ongoing and the energy and goodwill this has generated. However we plan to address an area of application that has not yet received much attention within the BEV: the development of business services that would be offered over the network (e.g., retail ordering, restaurant reservations and information, advertising, banking).

We propose to study the effectiveness of scenario-based participatory design sessions with members of the community (business service providers and their users). The methodology will be modeled on that described by Morten Kyng and his colleagues (Greenbaum & Kyng, 1991; Kyng, 1991; Kyng, 1995). Briefly, a design team surveys an existing usage situation to develop a pre-understanding of problems and issues that might be addressed by new technology. They then prepare "envisionment" materials for sharing with users in cooperative design workshops. A very rough scenario might serve as a starting point for discussion, but the ultimate design proposals are generated with participation from potential users. We plan to work on developing the initial workshop materials during the first summer of the grant period. We will then take advantage of an extended trip to Aarhus (to participate in two back-to-back conferences, one on object-oriented technology, the other on use-oriented design), to consult with Kyng and his colleagues regarding this study (see attached letter of support).

One outcome of these cooperative design sessions will be specifications and prototypes of new community network services. However, from the perspective of the current project, the more relevant outcome will be our increased understanding of the role that scenarios play in such a cooperative design setting. Our analysis of the design sessions (we will collect videotaped records of the group meetings as well as any artifacts produced) will focus on two major issues. First, we will examine how scenario representations aid or interfere with designer-user communication. We expect that in general these concrete representations will facilitate the exchange of ideas, but we plan to study in detail what aspects of the scenarios are most useful in creating and evolving a shared understanding, and which aspects get in the way of such cooperation.

A second question concerns the possible contributions of object-oriented modeling to cooperative design. Elsewhere we have suggested that the "active agent" metaphor that is

often evoked in reasoning about object responsibilities in OOD might help users to extend their own understanding of tasks (Rosson & Alpert, 1990). We plan to use these participatory design sessions to study this in a quasi-experimental fashion: the designers in some of the sessions will introduce and exemplify the object metaphor, and encourage users to apply it in thinking about the scenarios being developed. We will then study both the process of scenario generation and the resulting scenarios for differences that might be attributable to users' exposure to the object-oriented paradigm (including possible interactions with designer-user communication activities).

As conveyed by Figure 2, the cooperative design study will take place over an extended period of time (about a year and a half). This is partially due to the fact that finding the right problems and the right set of participants will take considerable time and effort. We also will need to do a fair amount of work in advance of the sessions, both to interview the business providers and their users, and to develop seed scenarios. Finally, an important prerequisite for these sessions will be an adequate set of cooperative prototyping tools. Our current plan is to develop a tool kit in Macromind Director that can be used to "program" prototypes cooperatively and in real time, during group sessions. Note that such a cooperative prototyping environment is itself a valuable project deliverable.

Scenario transformation. Later in the project, our analyses of scenario generation will be complemented by an investigation of how well these narrative descriptions can serve as input into other design activities. One of these activities — user-designer communication — will be studied as part of the cooperative design project described above. However, that study will take place in a real world design setting and the precision of the analysis will suffer from the complexities and incidental factors inherent in such settings. Thus we plan to complement that analysis with a more controlled laboratory study. At the same time, we will examine the effectiveness of scenarios as input into two other design activities, the development of a functional specification document, and the development of a high-level software design.

The study will take place in two phases. First, we will investigate the process of converting task scenarios into functional specifications, i.e., a well-ordered, non-redundant listing of the features exercised in the scenarios. Software designers will be asked to create organized specification documents from refined versions of scenarios generated in the earlier studies. Our analysis of this process (participants will work in pairs, and we will record their discussion and work products) will focus on how the designer participants address issues of completeness and consistency in working from the incomplete specification provided by the scenarios.

In the second phase of the study, we will merge the specification documents produced by the participants from the first phase, to create a single canonical specification. This specification, along with the original scenarios, will serve as the experimental materials for two further tasks. In one case, representative users will be provided with either the functional specification or the scenarios as a representation of a system's functionality. The prospective users' understanding will be "tested" through tasks requiring evaluation of the usage pros and cons of the proposed design, and the generation of "what if" situations.

The same experimental materials will be used for a parallel study of software design. Again working in pairs (with their discussion and notes recorded), software designers will be asked to develop a high-level design for the functionality described in either the functional specification or the scenario set. Some of the designers will be asked to produce an object-oriented design; others a conventional functional decomposition. An effort will be made to equate the functional and object-oriented groups as much as possible; however, prior design experience will be the most critical grouping factor. We will analyze the

problems and efficacies of working from the two different representations of requirements as a function of the kind of design being constructed.

Tools for scenario-based development. Throughout the three-year project, we will continue our prior research into tools for the support of a scenario-based design approach. As we obtain a better understanding of scenario generation and application, through our field work and laboratory studies, we will direct this understanding back into our work on support tools. Early on, for example, we will be developing tools for the cooperative prototyping of scenarios. As the project continues, we will continue the research initiated in the Scenario Browser project, developing an environment that integrates the design of task specification (scenario narratives) and implementation (object-oriented software).

The software development tools will be built in Smalltalk. A major reason for choosing Smalltalk is that it provides a highly dynamic language and development environment, which is essential for an approach that assumes considerable bottom-up development and refinement of software abstractions. Our work on the Scenario Browser used Digitalk's Smalltalk/V®. However, the Virginia Tech College of Engineering has recently decided to acquire a site license for ParcPlace Smalltalk/80, a much richer and even more dynamic Smalltalk implementation. Thus it is quite likely that the new work will shift to this alternate platform. The decision to acquire a license for Smalltalk/80 reflects a move in Engineering toward a greater emphasis on design issues, and on OOD issues in particular. Thus, a side benefit of shifting to the new platform will be the establishment of technical links between our research project and the faculty and students in Engineering working on issues of object-oriented design.

A new area of research addressed in the work on support tools will be collaboration in design. Using our observations of the cooperative design sessions, as well as the lab studies of designer pairs, we will experiment with support for team members' independent but coordinated contributions to a developing design.

Although many software developers are familiar with the concept of scenarios, and may even have created them as part of a requirements document, the notion of simultaneous development of task and software designs is a new one. As a result, it will be difficult to assess the long-term effectiveness of the tools that we develop: we would first need to "retrain" experienced designers in the scenario-based approach, and this is too time-consuming to carry out during the three-year period of this project. However, we can study its use by less experienced designers (e.g., computer science students first learning object-oriented software design). For example, we plan to carry out a formative evaluation of the tool through its use in a Special Topics course on object-oriented software engineering (see Planned Education Activities in Section 2.2). We expect that a side benefit of an environment supporting scenario-based design of object-oriented applications is that it might help inexperienced designers carry out and rationalize the difficult step of identifying objects and assigning responsibilities (Wirfs-Brock, Wilkerson, & Wiener, 1990).

1.4 Prior Research Accomplishments

Rosson has a strong research record in HCI. In collaboration with Carroll (the other senior investigator participating in the proposed project) and other colleagues at the IBM T. J. Watson Research Center, she developed the *task-artifact framework*, an "action science" view of HCI in which science and design knowledge are seen to evolve together. The framework is grounded in the constant interplay between tasks and the artifacts recruited for these tasks: tasks set requirements for designed artifacts, which in turn raise new opportunities for tasks. By analyzing, explaining and generalizing the consequences that artifacts have for users engaged in tasks, researchers can begin to build a cumulative science base. At the same time, by supporting a more articulated reasoning process — one that attempts to mitigate negative consequences and exploit positive ones — such analyses

can contribute to the design of better artifacts. Tasks are the fundamental unit of analysis in this framework, hence our interest in scenario-based design methods. Our presentation of this framework has covered its philosophical and theoretical foundations (Carroll & Campbell, 1989; Carroll & Kellogg, 1989), as well as methods for its application to projects in science and design (Carroll, Kellogg, & Rosson, 1991; Carroll & Rosson, 1992).

The task-artifact framework has been developed and communicated through a number of design case studies complemented with empirical evaluation. Most of these have been in the domain of software technology, with emphasis on the design of tools for learning and using the Smalltalk programming language and environment (Carroll & Rosson, 1992; Carroll, Rosson, & Singley, 1993; Carroll, et al., 1991; Rosson & Carroll, 1993a; Rosson & Carroll, 1993b; Rosson, Carroll, & Bellamy, 1990; Rosson, Carroll, & Sweeney, 1991). The work on the Scenario Browser summarized in Section 7.1.2 represents a melding of the task-artifact research program with Rosson's research on tools for instance-centered design in Smalltalk (Gold & Rosson, 1991). Rosson's work on the learning and use of object-oriented technology has earned her a unique position of external visibility within both the HCI and object-oriented research communities.

Rosson has also carried out several large survey studies, of text-editor usage patterns (Rosson, 1984), of software development practices (Rosson, et al., 1990), and of user interface programming (Myers & Rosson, 1992). This work provides valuable methodological experience in support of the scenario survey study.

2.0 EDUCATION PLAN

I have considerable experience in studying and developing education. However, my experience came through research on professional training issues. My challenge now is to take that experience and apply it to my new environment of academic computer science. Many of the education activities described here are designed to support this transition. The activities are also designed to assist the Computer Science Department in enhancing its program in HCI, and to contribute to Virginia Tech's current focus on developing innovative and cross-disciplinary course offerings.

2.1 Objectives of Education Activities

Doing user-centered design is hard. If we as researchers in HCI expect software development teams to succeed at this, then we had better provide tools and techniques that make it easy and fun. We had also better consider the development process as a whole: well-intentioned and user-centered decisions made during the gathering of requirements can easily be lost as the team moves toward a software realization of these requirements. These beliefs have driven me to a research program that is oriented toward the design *process* and toward *tools* to support that process.

Of equal importance to the development of use-oriented design methods and tools is the training of software developers to use these tools. A prime place to carry out such training is in a university setting, where students are acquiring the knowledge and skills that will form the foundation of their professional lives. A major motivation for me in making the switch from industrial research (at IBM's T.J. Watson Research Center) to an academic position was the opportunity it gives me to influence students' emerging views of how interactive systems should be developed. Thus an important general goal for me is to find ways to embed my research findings and activities into university education contexts.

I was joined the Computer Science Department of Virginia Tech at a time when it was increasing its emphasis on HCI: like many similar departments, we are struggling to define a position for this subarea as pat of a well-rounded undergraduate computer science curriculum (see, e.g., the recent NSF report on "New Directions in Human-Computer Interaction). As a result, I have the opportunity — indeed the responsibility — to develop several new courses over the next few years. Developing and refining these courses will teach me a great deal about education in a university environment. Making them a standard part of our CS curriculum will require a careful integration of HCI with other subareas (e.g., software engineering). And because the courses will focus to a great extent on design process, I will be able to fulfill one of my major goals in taking this university position: where appropriate, I will bring into the courses the issues and methods I am examining in my research on scenario-based design. This will provide valuable feedback to me and my research colleagues, as well as exposing students to state-of-the-art research in design methods.

For the most part, the courses I plan to develop will be directed toward CS majors. However, the interdisciplinary nature of HCI raises an opportunity for considerable educational "outreach" to related disciplines within the university (e.g., education, psychology, anthropology, management, engineering, technology studies). Developing cross-disciplinary courses is quite consistent with the current educational climate at Virginia Tech, which like many universities, is searching for innovative ways to provide richer course offerings at a reasonable cost to faculty, students, and state funding sources.

I am also committed to "educating" HCI colleagues and practitioners working on the design of interactive systems. Thus, I will continue to communicate our emerging ideas concerning scenario-based design to practitioners and researchers — in both the HCI and

OOD communities — through conference presentations, publications, tutorials and short courses.

Another aspect of my education objectives concerns the Blacksburg community. As an HCI researchers studying the BEV community network project, I have been struck by the tremendous cultural gap between the network-savvy and the rest of the community: language barriers, network communication "etiquette", technology expectations. One reason I have chosen the cooperative design approach — in addition to its effectiveness at producing useful, usable applications — is the educative impact it will have on the participating community members. Through joint envisionment of new network-based task scenarios, complemented with discussion of the design issues raised by these scenarios, participants will gain a better appreciation of the opportunities and concerns raised by this technology likely to be pervasive in their lives in the near future.

2.2 Planned Education Activities

Departmental curriculum planning. As a newly elected member of our department's Undergraduate Program Committee, I am now and will continue to participate in the re-vamping of our CS majors requirements to admit a greater focus on design process and specifically on HCI issues in design. It is not enough to develop and offer new courses covering HCI material, we must also make "room" in students' schedules to take these courses. Part of this work involves educating co-faculty on how to think about HCI as a subarea of computer science; just as important is my learning enough about other CS subareas to see where our issues and concerns fit in.

The current departmental climate is very favorable toward expanding the coverage of HCI as a sub-discipline. Through the work of Rex Hartson, Deborah Hix, and Roger Ehrich (along with Robert and Beverly Williges, their collaborators from Industrial Engineering), our department established one of the earliest academic research programs in HCI. These investigators (with Ed Fox, a highly visible researcher in multimedia and digital libraries) were recently awarded an NSF Research Infrastructure grant. At the same time, another group of faculty were awarded an NSF Education Infrastructure grant. These two grants are now reinforcing each other just as one might hope: as a department, we have begun to explore innovative interaction technology as a vehicle for innovative education technology. I am in the fortunate position of leveraging the attention and interest now being devoted to HCI issues within the department, both with respect to the education plans described here and the support of the research described in Section 1.0.

Refinement of existing courses. I have taught one course since arriving in January 1994, "Computing in the Liberal Arts". The course is an overview of Computer Science concepts and approaches for non-majors. My version of the course introduced exercises in which students developed discipline-specific scenarios exemplifying concepts currently under discussion. These exercises were quite successful, and future versions of the course (I am scheduled to teach it again in Spring 1995) will expand the role of such scenario-based envisionment. Similarly, at some point over the next three years, I am likely to teach the graduate HCI course, a usability methods course in which teams of students design, prototype and evaluate an interactive application. When I teach that course, scenarios will be introduced as an important and useful design representation, and scenario generation and evolution will be a component of the usability methods exercised by the class.

New courses. I expect to develop three new courses over the next three years. First, in keeping with the department's expanded emphasis on HCI, I have collaborated with another faculty member (Rex Hartson) in proposing our first undergraduate HCI course, a junior-level course "Introduction to Human-Computer Interaction." This course is likely to be offered for the first time in Spring 1996; I will develop and teach it. The course is not a lab course (i.e., students will not build systems), but I plan to provide a strong design

perspective. It is intended for both CS majors and upper-division students from other disciplines. My vision for the course is to get majors and non-majors working together, such that the computer science students learn to discover and respond to users' needs and concerns, while the non-majors learn to recognize and express their needs to software developers.

A second course is planned as an initial extension to our graduate HCI curriculum (we currently offer just the one methods course, which is very popular). This course will cover the general area of computer-supported cooperative work. As part of the course, students will analyze their own collaborative activities in designing and prototyping a cooperative work application. Scenario-based envisionment and participatory design will be offered as useful techniques in support of students' design projects. I expect this course to be first offered in the Fall of 1995 or 1996.

The third planned course will be a special topics course developed for senior-level CS majors, "Object-oriented Software Engineering with Use Cases". Students will apply the use-case concept (a use case is somewhat like an "abstract" scenario, in that it covers a use transaction but includes all possible variants) developed by Ivar Jacobson and his colleagues (Jacobson, et al., 1992) to development of an interactive system. Development will be done in Smalltalk-80 and students will use the tools developed as part of my research on scenario-based design. Formative evaluation of the tools will be carried out in this class, using as data students' comments, design logs and documentation, as well as the quality of the applications they produce.

Interactive tutorial materials for Smalltalk. I have considerable experience in developing interactive learning environments for Smalltalk, and this experience will be applied to the development of materials suitable for undergraduates or graduates interested in learning about or using Smalltalk. As in past versions, these will be self-study materials, so they should be useful to students taking classes for which they have to use Smalltalk, as well as to students (or faculty) who have a "passing interest" in this particular language. I have already begun a new version (with Scott Atha, a student working on an undergraduate research project), for Smalltalk/V on the Macintosh. Once a site license to Smalltalk-80 has been acquired by the College of Engineering, work will commence on a version for that platform.

Professional education. Carroll and I have collaborated in the past on tutorials for CHI and OOPSLA, and we plan to continue making these contributions. For example, we have recently had a new tutorial accepted for presentation at CHI'95: "Introduction to Object-oriented Design: A Minimalist Approach". This tutorial exploits HCI practitioners' familiarity with task scenarios as a vehicle for introducing the basic concepts of object-oriented analysis and design. The demonstrations and exercises rely heavily on our existing Scenario Browser environment. Once this tutorial has been prepared, we also expect to offer it as a "short course" periodically at Virginia Tech.

Community education. Although we do not plan to set up community education courses specifically directed at expanding community members' view of network-based applications, our cooperative design sessions will have this as a side-effect. In addition to the specific learning that occurs through the shared design work, this project will also establish us as a contact point for community members wishing to learn more about network applications, and we expect to provide ongoing informal education through these channels.

2.3 Planned Teaching Activities

The normal teaching load in our department is 3 courses per year. Over the next three years, I expect that my personal load will include 2-3 iterations of both the new undergraduate HCI course and "Computing as a Liberal Art". In addition, I anticipate

teaching the CSCW course described above and the special topics course on OOD. The remaining course load will probably include the current graduate HCI course, as well as other undergraduate courses which will be new to me (e.g., professionalism, computer literacy).

2.4 Summary of Education Accomplishments

I have only just begun my academic teaching career: I taught two sections of "Computing as a Liberal Art" during the Spring 1994 term. However, I have considerable research experience in the area of education. During my tenure at IBM, I collaborated with Carroll in his development and application of the Minimalist model of instruction (Carroll, 1990). The learners in that work were professionals rather than university students; however, the model's constructs are general, and one of our current goals is to adapt the "active learning" approach of minimalism to more conventional classroom-based instruction.

My most recent — and most intensive— work on professional education has been in the area of object-oriented programming and design, with an emphasis on the Smalltalk language and environment. Over the past five years, I collaborated with Carroll and others on the design and implementation of minimalist interactive learning tools for experienced procedural programmers encountering Smalltalk for the first time. These tools formed the basis of a self-study tutorial (MiTTS, Minimalist Tools and Training for Smalltalk), which was distributed to more than 30 development labs within IBM, as well as to a small number of IBM customer sites. One of our recent tutorials (at OOPSLA'93) offered MiTTS as an example of minimalist instruction in OOP/OOD, and challenged professors to join us in taking a more active, minimalist approach to teaching this material in a university setting. Just prior to leaving IBM, Carroll and I collaborated with others (Robertson, Mack, Koenemann-Belliveau, Alpert) on yet another minimalist learning environment (ODE, Object Design Exploratorium, a paper on this system will be presented this year at OOPSLA). Unlike MiTTS, this more recent project was language-independent and directed more explicitly at object-oriented design issues.

Other professional tutorials taught at professional meetings include several presentations of the task-artifact framework (with Carroll at a regional SIGCHI meeting, and at the British HCI Summer School in 1990; with Carroll and Singley at CHI'92), and an overview of Minimalism with examples (with Carroll at CHI'90).

At an informal level, I am currently participating in a graduate-level special topics course "Scenario-based Design". The course relies extensively on the edited proceedings of a scenario workshop sponsored by IBM in the summer of 1993, to which I contributed a paper, as well as on a book on scenario-based design currently under preparation by Carroll and myself. I have given several guest lectures, as well as participating in class discussions on a regular basis.

Finally, I have taken an active role in promoting education within the OOPSLA professional community. Along with Jim Heliotis of Rensalaer Polytechnic Institute, I organized and chaired the first OOPSLA Educators' Symposium (in 1992); this event provided a forum for computer science educators to share their experiences and concerns regarding the education of students in object-oriented techniques. The symposium was very successful and led to an invited special issue of the journal *Computer Science Education.* It also established me as a visible member of the CS education community, even prior to my transition from IBM researcher to CS faculty member. This year, I am organizing and chairing the first Doctoral Symposium at OOPSLA, a closed workshop in which a small set of Ph.D. students discuss their thesis work with a panel of experts and fellow students.

3.0 DEPARTMENTAL ENDORSEMENT

Effective date of appointment of Mary Beth Rosson to the position of Associate Professor of Computer Science: January 1, 1994.

"I have read and endorse this Career Development Plan." (Because of the personal relationship between John Carroll and Mary Beth Rosson, Rosson's personnel manager at Virginia Tech is Michael Williams, Director of the Computing Center).

_____ _____

John M. Carroll Date

Department Head, Computer Science

_____ _____

Michael Williams Date

Director, Computing Center

D. BIBLIOGRAPHY

Campbell, R. L. (1992). Will the real scenario please stand up? *SIGCHI Bulletin*, *24*(2), 6-8.

Carroll, J. M. (1990). *The Nurnberg Funnel: Designing Minimalist instruction for practical computer skill*. Cambridge, MA: MIT press.

Carroll, J. M., & Campbell, R. L. (1989). Artifacts as psychological theory: The case of human-computer interaction. *Behavior and Information Technology*, *8*, 247-256.

Carroll, J. M., & Kellogg, W. A. (1989). Artifact as theory-nexus: Hermeneutics meets theory-based design. In K. Bice & C. Lewis (Ed.),^(Eds.), *Proceedings of CHI'89: Human Factors in Computing Systems*, (pp. 7-14). New York, NY: ACM.

Carroll, J. M., Kellogg, W. A., & Rosson, M. B. (1991). The task-artifact cycle. In J. M. Carroll (Ed.),^(Eds.), *Designing Interaction: Psychology at the Human-Computer Interface* (pp. 74-102). New York: Cambridge University Press.

Carroll, J. M., & Rosson, M. B. (1990). Human-computer interaction scenarios as a design representation. In *Proceedings of HICSS-23: Hawaii International Conference on System Sciences*, (pp. 555-561). Los Alamitos, CA: IEEE Computer Society Press.

Carroll, J. M., & Rosson, M. B. (1991). Deliberated evolution: Stalking the View Matcher in design space. *Human-Computer Interaction*, *6*, 281-318.

Carroll, J. M., & Rosson, M. B. (1992). Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Transactions on Information Systems, 10*, 181-212.

Carroll, J. M., Rosson, M. B., & Singley, M. K. (1993). The Collaboration thread: A formative evaluation of object-oriented education. In *Proceedings of Fifth Workshop on Empirical Studies of Programming*, . Norwood, NJ: Ablex.

Carroll, J. M., Singley, M. K., & Rosson, M. B. (1991). Integrating theory development with design evaluation. *Behavior and Information Technology*, *11*(5), 247-255.

diSessa, A. A. (1991). Local sciences: View the design of human-computer systems as cognitive science. In J. M. Carroll (Ed.),^(Eds.), *Designing Interaction: Psychology at the Human-Computer Interface* (pp. 162-202). Cambridge: Cambridge University Press.

Ehn, P. (1988). *Work-Oriented Design of Computer Artifacts*. Hillsdale, NJ: Erlbaum.

Gibson, E. (1990). Objects — Born and bred. *Byte*, 245-254.

Gold, E., & Rosson, M. B. (1991). Portia: An instance-centered environment fo Smalltalk. In *Proceedings of OOPSLA'91: Object-oriented Programming Systems, Languages, and Applications*, (pp. 62-74). New York, NY: ACM.

Greenbaum, J., & Kyng, M. (1991). *Design at Work: Cooperative Design of Computer Systems*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Harrison, M., & Thimbleby, H. (Ed.).^(Eds.). (1989). *Formal Methods in Human-Computer Interaction*. New York, NY: Cambridge University Press.

Jacob, R. J. K. (1983). Executable specifications for a user interface. In A. Janda (Ed.),^(Eds.), *Proceedings of CHI'83: Human Factors in Computing Systems*, (pp. 28-34). New York, NY: ACM.

Jacobson, I., Chriserson, M., Jonsson, P., & Overgaard, G. (1992). *Object-oriented Software Engineering: A Use-case Driven Approach*. Reading, MA: Addison-Wesley.

Koenemann-Belliveau, J., Carroll, J. M., Rosson, M. B., & Singley, M. K. (1994). Comparative usability evaluation: Critical incidents and critical threads. In C. Plaisant

(Ed.),^(Eds.), *Proceedings of Human Factors in Computing Systems, CHI '94 Conference*, (pp. 245-251). New York: ACM.

Kyng, M. (1991). Designing for cooperation — cooperating in design. *Communications of the ACM, 34*(12), 64-73.

Kyng, M. (1995). Creating contexts for design. In J. M. Carroll (Ed.),^(Eds.), *Scenario-based Design: Envisioning Work and Technology in System Development* New York, NY: John Wiley.

Myers, B. A., & Rosson, M. B. (1992). Survey on user interface programming. In P. Bauersfield, J. Bennett, & G. Lynch (Ed.),^(Eds.), *Proceedings of CHI'92: Human Factors in Computing Systems*, (pp. 195-202). New York, NY: ACM.

Reitman, W. (1965). *Cognition and Thought*. New York: John Wiley.

Roberts, T. L., & Moran, T. P. (1983). The evaluation of text editors: Methodology and empirical results. *Communications of the ACM, 26*, 265-283.

Rosch, E., Mervis, C. B., Gray, W., Johnson, D., & Boyes-Braem, P. (1976). Basic objects in natural categories. *Cognitive Psychology, 7*, 573-605.

Rosson, M. B. (1984). Effects of experience on learning, using, and evaluating a text-editor. *Human Factors, 26*, 463-475.

Rosson, M. B., & Alpert, S. R. (1990). Cognitive consequences of object-oriented design. *Human-Computer Interaction, 5*, 345-379.

Rosson, M. B., & Carroll, J. M. (1993a). Active programming strategies in reuse. In O. M. Nierstrasz (Ed.),^(Eds.), *Proceedings of ECOOP '93 — Object-Oriented Programming*, (pp. 4-18). Berlin: Springer-Verlag.

Rosson, M. B., & Carroll, J. M. (1993b). Extending the task-artifact framework: Scenario-based design of Smalltalk applications. In H. R. Hartson & D. Hix (Ed.),^(Eds.), *Advances in Human-Computer Interaction* (pp. 31-57). Norwood, NJ: Ablex.

Rosson, M. B., & Carroll, J. M. (1995). Narrowing the specification-implementation gap in scenario-based design. In J. M. Carroll (Ed.),^(Eds.), *Scenario-based Design: Envisioning Work and Technology in System Development* New York, NY: John Wiley.

Rosson, M. B., Carroll, J. M., & Bellamy, R. K. E. (1990). Smalltalk scaffolding: A case study in Minimalist instruction. In J. C. Chew & J. Whiteside (Ed.),^(Eds.), *Proceedings of Human Factors in Computing Systems, CHI '90 Conference*, (pp. 423-429). New York, NY: ACM.

Rosson, M. B., Carroll, J. M., & Sweeney, C. (1991). A View Matcher for reusing Smalltalk classes. In *Proceedings of Human Factors in Computing Systems (CHI'91)*, (pp. 277-284). New York, NY: ACM.

Rosson, M. B., Maass, S., & Kellogg, W. A. (1990). The designer as user: Building requirements for design tools from design practice. *Communications of the ACM, 31*(11), 1288-1297.

Rubin, K. S., & Goldberg, A. (1992). Object behavior analysis. *Communications of the ACM, 35*(9), 48-62.

Swartout, W., & Balzer, R. (1982). On the inevitable intertwining of specification and implementation. *Communications of the ACM, 25*(7), 438-445.

Taichritzis, D., Fiume., E., Gibbs, S., & Nierstrasz, O. (1987). KNOs: Knowledge acquisition, dissemination, and manipulation objects. *ACM Transactions on Office Systems, 5*, 96-112.

Wasserman, A. I. (1982). Rapid prototyping of interactive information systems. *ACM Software Engineering Notes, 7*, 171-180.

Wirfs-Brock, R., Wilkerson, B., & Wiener, L. (1990). *Designing Object-oriented Software*. Englewood Cliffs, NJ: Prentice Hall.

Mary Beth Rosson

Associate Professor of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061-0106
office: 703-231-6470; fax: 703-231-6075; email: rosson@vt.edu

Education:

Ph.D., Human Experimental Psychology, University of Texas, Austin, Texas (1982)

B.A., Psychology, Trinity University, San Antonio, Texas (1977)

Honors:

Who's Who of American Women (1992)

Phi Beta Kappa; Trinity University (1976)

Employment History:

Associate Professor, Virginia Polytechnic Institute and State University (1994-present)

Research Staff Member, IBM T.J. Watson Research Center (1982-1993)

Human Factors Psychologist, Development Human Factors, IBM Austin (1981)

Professional Service:

CHI (the annual conference of ACM SIGCHI): Panels Chair, 1989; Program Committee, 1985, 1988, 1991, 1992; Associate Papers Chair, 1994; Papers Chair, 1995

OOPSLA (Object-oriented Programming Systems, Languages and Applications): Educators' Symposium Co-Chair, 1992; Program Committee, 1994, 1995; Doctoral Symposium Chair, 1994

UIST (User Interface Software and Technology): Panels, 1989; Program Committee, 1990

Human Sciences Editorial Board, *Interacting with Computers* (1989-present)

Recent publications related to the current project:

Note: Rosson and Carroll have published many papers jointly; however, the two bibliographic sketches include no overlap in cited publications.

Carroll, J. M. & Rosson, M. B. 1992. Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Transactions on Information Systems, 10* (2), 181-212.

Rosson, M. B. and Alpert, S. R. 1990. The cognitive consequences of object-oriented design. *Human-Computer Interaction, 5*, 345-379.

Rosson, M. B., Carroll, J. M. and Bellamy, R. K. E. 1990. Smalltalk scaffolding: A case study in Minimalist instruction. In J.C.Chew & J.Whiteside (Eds.,) *Human Factors in Computing Systems, Proceedings of CHI'90* (pp. 423-430). New York: ACM.

Rosson, M. B. and Carroll, J. M. 1993. Extending the task-artifact framework. In R.Hartson & D.Hix, (Eds.), *Advances in Human-Computer Interaction, Volume 4* (pp. 31-57).

Rosson, M. B. and Carroll, J. M. 1995. Narrowing the gap between specification and implementation in scenario-based design. In J.M.Carroll (Ed.), *Scenario-based Design: Envisioning Work and Technology in Software Development*. New York: John Wiley (in press).

Other significant recent publications:

Gold, E. and Rosson, M. B. 1991. Portia: An instance-centered environment for Smalltalk. In *Proceedings of OOPSLA'91* (pp. 62-74). New York: ACM. (October 6-10, Phoenix).

Koenemann-Belliveau, J., Carroll, J. M., Rosson, M. B., and Singley, M. K. 1994. Comparative usability evaluation: Critical incidents and critical threads. In *Proceedings of CHI'94* (pp. 245-251). New York: ACM Press.

Carroll, J. M., Rosson, M. B. and Singley, M. K. 1993. The collaboration thread: A formative evaluation of object-oriented education. In *Proceedings of the Fifth Workshop on Empirical Studies of Programmers*. Norwood, NJ: Ablex.

Rosson, M. B. and Carroll, J .M. 1993. Active programming strategies for reuse. In *Proceedings of ECOOP'93*. Springer-Verlag. (Kaiserslautern, Germany, 26-30 July).

Rosson, M. B., Maass, S., and Kellogg, W. A. 1989. The designer as user: Building requirements for design tools from design practice. *Communications of the ACM, 31* (11),1288-1297.

Significant recent collaborators not mentioned in publications:

John Karat, Robert Mack, Scott Robertson, Christine Sweeney, Mary van Deusen

Graduate advisor:

Philip Gough

John M. Carroll

Professor of Computer Science and Psychology
Computer Science Department Head
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061-0106
office: 703-231-6931; fax: 703-231-6075; email: carroll@cs.vt.edu

Education:

Ph.D. Psychology, 1976, Columbia University, New York, N Y.

B.A. Mathematics; Information Sciences (1972); Lehigh University, Bethlehem, Pa.

Honors (past decade):

Association for Computing Machinery (ACM), Recognition of Service Award, 1987.

ACM Special Interest Group on Documentation, Rigo Career Achievement Award, 1994.

Employment History:

Professor and Head, Computer Science, Virginia Polytechnic Institute and State University (1994-present)

Research Staff Member and Manager, IBM T.J. Watson Research Center (1976-1994)

Editorial Board Memberships:

International Journal of Man-Machine Studies (since 1985),
Behaviour and Information Technology (since 1986),
International Journal of Human Computer Interaction (founding member, 1988),
Le Travail Humain (since 1990),
Cambridge University Press Human-Computer Interaction Series (since 1990),
Human-Computer Interaction (since 1990),
ACM Transactions on Computer-Human Interaction (founding Associate Editor, 1992),
Ablex Press Advances in Human-Computer Interaction Series (since 1994),
Wiley Handbook of Human-Computer Interaction (since 1994).

Recent publications related to the current project:

Note: Rosson and Carroll have published many papers jointly; however, the two bibliographic sketches include no overlap in cited publications.

Carroll, J. M. 1994. Making use a design representation. *Communications of the ACM, 37,* in press.

Carroll, J. M. 1994. Designing scenarios for human action. *Performance Improvement Quarterly, 7/3,* 64-75.

Carroll, J .M., Mack, R. L., Robertson, S. P. & Rosson, M. B. 1994. Binding objects to scenarios of use. *International Journal of Human-Computer Studies , 40,* (formerly, International Journal of Man-Machine Interaction) in press.

Carroll, J. M. 1993. Creating a design science of human-computer interaction. *Interacting with Computers, 5/1,* 3-12.

Carroll, J. M., Koenemann-Belliveau, Rosson, M .B. & Singley, M. K. 1993. Critical incidents and critical threads in empirical usability evaluation. In J. Alty, D. Diaper & S.P. Guest (Eds.), *People and Computers VIII, Proceedings of the HCI'93 Conference.* Cambridge: Cambridge University Press, 279-292.

Other significant recent publications:

Carroll, J. M. (Ed). 1995. Scenario-based design.: Envisioning work and technology in system development. New York: John Wiley and Sons (in press).

Carroll, J. M. (Ed). 1991. Designing Interaction: Psychology at the human-computer interface. New York: Cambridge University Press.

Carroll, J. M. 1990. The Nurnberg Funnel: Designing minimalist instruction for practical computer skill. Cambridge, MA: MIT Press.

Carroll, J. M. and Rosson, M. B. 1991. Deliberated evolution: Stalking the View Matcher in design space. Human-Computer Interaction, 6, 281-318.

Carroll, J.M., Rosson, M.B. and Singley, M.K. 1993. The collaboration thread: A formative evaluation of object-oriented education. In *Proceedings of the Fifth Workshop on Empirical Studies of Programmers*. Norwood, NJ: Ablex.

Significant recent collaborators not mentioned in publications:

Sherman Alpert, Rachel Bellamy, Robert Campbell, John Karat, Wendy Kellogg, Clayton Lewis, Jean McKendree, Tom Moran, John Richards, Mary van Deusen, Hans van der Meij

Graduate advisor:

Thomas Bever

F. BUDGET JUSTIFICATION

Personnel. The personnel costs reflect one month of summer support for each of the three years for the principle investigator, Mary Beth Rosson. In support of the empirical work and some of the tool development, it also includes one GRA for each of the three years. In support of the programming and the coding of data, undergraduate wages are requested. In support of Rosson's research and education goals, the Computer Science Department will include 10% of her academic year salary in cost-sharing (see attached cost-sharing documentation).

Fringe benefits. We request the standard Virginia Tech percent (4% for faculty summer salary).

Travel. For each of year of the grant, $2500 domestic travel is requested. This is based on a plan to attend two conferences each year (the annual CHI and OOPSLA conferences), and will be supplemented as necessary and possible with travel funds from the department and college. Rosson is one of the few members of the HCI community who has established a reputation in both research communities represented by these conferences, and regular participation in the conferences is critical to her continued development in this regard.

The year 1 budget includes an additional $1000 in foreign travel funds. These funds will enable Rosson to exploit the fortuitous contiguous occurrence of two important conferences at the University of Aarhus in Denmark: ECOOP'94, followed by Computers in Context. ECOOP is the major non-US conference in object-oriented technology; Computers in Context is one of the first international conferences to focus explicitly on the relationship between computer use and computer design. Both conferences are directly relevant to the research and education projects described in the current proposal. Furthermore, Rosson will use this 2-3 three week visit to Aarhus (in August 1995) to consult with colleagues in the Computer Science Department who have extensive experience in the area of cooperative design (Morten Kyng, Susanne Bødker). At this point in the project, initial plans will have been laid for the cooperative design sessions, and these colleagues have agreed to review and provide input to these plans (see attached letter of support).

Materials and supplies. The budget includes $500 for each year for incidental supplies, e.g., tapes, diskettes, paper, mailing costs.

Computing services. The Computer Science Department requires that all principle investigators on grants purchase a computer account at the rate of $4000 per year. Because the CAREER program allows only a modest budget, and because it is directed toward faculty development, the department has waived all but $500 of this cost.

Tuition. Virginia Tech now requires that all GRA stipends include full tuition.

Software. The software charges are those associated with both the empirical and tool development work: copies of Macromind Director, a database program to manage the qualitative data, copies of Smalltalk and Smalltalk tools, other multimedia presentation systems, multivariate analysis packages. The requested amount is somewhat larger for the first year, as we will be carrying out much of our initial tool-building during that period.

Subject payments. Given the intensive nature of the laboratory studies planned, we will need to offer remuneration to the users and designers who participate. These charges will be greatest during the second year, when the multiple related scenario transformation experiments will be carried out. During the last year, the only evaluation activities planned will take place in a classroom setting and thus require no extra resource.

H. FACILITIES, EQUIPMENT AND OTHER RESOURCES

An important resource for the research component of the proposal will be the new laboratory facilities and equipment provided through the NSF Research Infrastructure grant to HCI researchers Ehrich, Fox, Hartson, Hix and Williges. These facilities will include a variety of research settings appropriate for the observational lab studies. The lab will also included a well-equipped video conferencing room which may be used in some of the cooperative design sessions with Blacksburg community members. An integral part of the lab will be a state-of-the-art video capture and editing environment, which will be invaluable in the analysis of the cooperative design workshops.

The planned education activities will also leverage the HCI Research Infrastructure facilities, in that many of the student design projects will be carried out using the equipment and software provided by these funds.

The cooperative design sessions devoted to BEV services will rely extensively on the BEV infrastructure already in place (i.e. network connections, on-line information and services).

I. SPECIAL INFORMATION AND SUPPLEMENTARY DOCUMENTATION

In support of this development plan, the Computer Science Department will cost-share 10% of Rosson's academic year salary. They will waive $3500 of the standard $4000 per annum charge for principle investigator's computer services. The laboratory space allotted to the HCI Research Infrastructure grant will be freely available for conduct of the research. Supplemental travel funds (up to $1000) will be available for travel in support of the development plan.