Educating Multi-disciplinary Design Teams

WENDY E. MACKAY

INRIA FUTURS[1]

LRI - BATIMENT 490, UNIVERSITE PARIS-SUD

91405 ORSAY CEDEX - FRANCE

Abstract

Designing interactive systems requires diverse expertise, which is why most successful design teams are multi-disciplinary. Unfortunately, managing such teams can be difficult, because team members often do not communicate effectively with each another. When we teach interaction design, we address this problem explicitly, with a two-fold approach: First, we explain the value systems and some of the key assumptions from the component disciplines, including social sciences, engineering and design. Second, we teach hands-on techniques, often with video, that place team members (and users) on an equal footing when expressing design ideas. We want our students to understand and respect the contributions of others outside their discipline and to be able to use design techniques that allow all team members to actively participate, whether observing users, generating new ideas, prototyping systems or evaluating them.

Keywords: Multidisciplinary Design, Science, Engineering, Design

[1] Projet In Situ, Pôle Commun de Recherche en Informatique du Plateau du Saclay, CNRS, Ecole Polytechnique, INRIA, Université de Paris-Sud.

Introduction

Designing interactive software is complex, requiring an understanding of human beings, software systems and the interaction between the two. Understanding people involves input from at least three social sciences. Psychology explores how the human sensory motor, perceptual and memory systems work, Sociology explores how people interact with each other, and Anthropology explores how people operate in the context of their daily activities. Developing interactive software also requires input from software engineering, including system architecture, programming languages, interaction techniques, as well as distributed computing and the use of a wide variety of hardware input and output devices. Creating innovative and aesthetically-pleasing designs requires input from trained designers, including graphic or interaction design and increasingly architecture and industrial design.

No single discipline provides all the necessary expertise: designing interactive software requires a multi-disciplinary approach. However, forming and managing multi-disciplinary teams has its own problems. Someone trained exclusively in one of the necessary disciplines is likely to interpret the design problem from within the framework of that discipline. This causes problems when people from different disciplines use the same words to mean different things or use different words to mean the same thing. As Djkstra-Erikson et al. [5] point out, "design" itself is a particularly troublesome word. Designers can only effectively communicate what they mean when they talk about the design *of* something: whether it is of the user experience, the screen layout, or the software architecture.

Another problem is that different disciplines place different values on different aspects of design. Scientists are trained to seek explanations of existing phenomena, engineers are trained to provide technical solutions to well-defined problems, and designers are trained to explore a design space and find solutions that "work". When people from these different backgrounds come together, they often run into conflicts due to their lack of a shared definition the problem.

Of course communication problems are not restricted to cross-disciplinary teams. For example, although research scientists share some common characteristics when compared to engineers or designers, when compared to each other, we also see different

priorities research methods. An experimental Psychologist who runs laboratory experiments values reliability and precision in the data. An anthropologist who studies people in field settings values context and the validity of the data.

Designers too operate with different priorities. For example, if you ask a book designer, a video producer and a photographer to design the layout of a screen, they will choose different focal points of attention. A book designer is trained to emphasize text organised in a grid and "knows" that a reader will look for the most important information in the upper left-hand corner. A video producer understands the aspect ratio and visual quality of video and "knows" that the center of the screen is the hot spot. A photographer used to the flexible aspect ratio of film and the fine gradations in visual quality will consciously avoid the center and will placement of key items along diagonal across the screen. Of course, any individual designer will deviate from these design principles for any particular design. What is important to understand for us to understand is that these designers are starting from different underlying principles: when they break rules, they are breaking different rules. When these rules are not stated explicitly, other team members are likely to other designers processes and solutions.

Component Interaction Design Disciplines

If we are to teach people to successfully participate in multi-disciplinary design teams, we must go beyond the explicit content of each discipline. Students need to learn about the diverse underlying value systems of relevant disciplines and reflect upon how they interact at a meta level.

Figure 1 shows some of the different disciplines that contribute to effective interactive system design. The three primary contributors derive from the social sciences, computer engineering, and design. From the natural sciences, we commonly find contributions from experimental Psychology (usually Cognitive Psychology, but increasingly Ecological Psychology and Activity Theory), as well as Sociology, Anthropology (particularly Ethnomethodology) and Human Factors or Ergonomics. From these social sciences, we borrow research findings, such as how people perceive information or how human memory works, as well as research techniques, such as how to run controlled experiments or conduct observational studies in the field.

Designers who use research techniques from any of these scientific disciplines must distinguish between their use in a purely scientific context and as a resource to support design. The underlying assumptions surrounding how these techniques are used, and the goals of the research, may differ greatly.

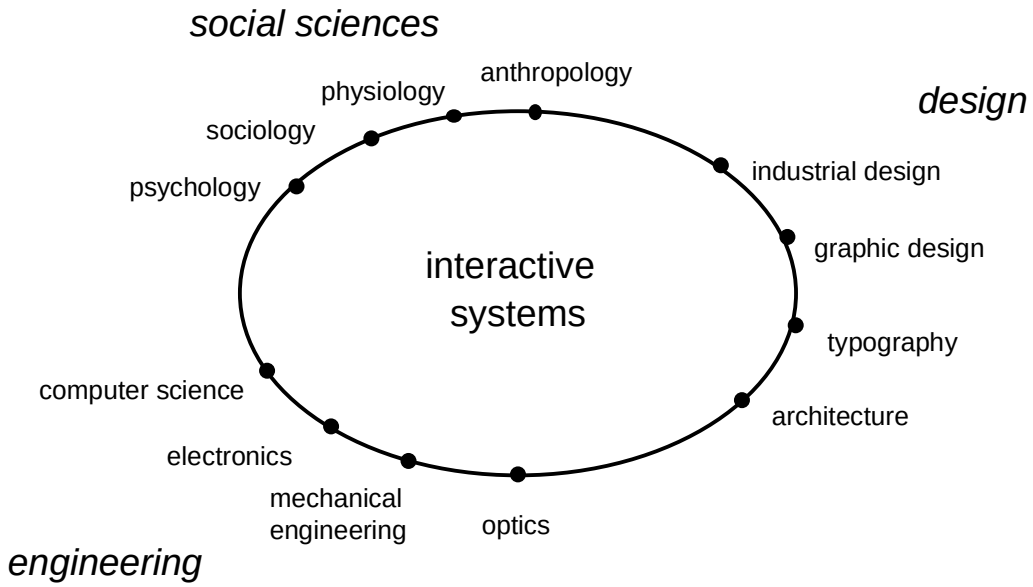

Figure 1: Interaction Design requires input from science, engineering and design disciplines.

For example, a usabilty study is not the same as a Psychology experiment. In experimental Psychology, the goal is to learn about fundamental characteristics of human beings, which exist independently of the experimenter. Controlled experiments are performed to test theories of human behavior, with the idea that they can be replicated by other researchers, who will then challenge or support the theory with further experiments. In contrast, usability studies are designed to evaluate particular software systems. Sometimes, the system is compared to another system, but the studies are rarely fully controlled in the scientific sense. The purpose is not to test theories of human behavior but rather to find problems with the system that was built and to test the adequacy of a particular design solution. Usability studies are rarely performed with the idea that they will be replicated and extended, but usually stand alone. A usability study is considered successful if it offers concrete information about the success of the particular system design for a particular set of users, but need not contribute the our general understanding of human beings.

Similarly, HCI professionals are careful to distinguish between *ethnography* and *ethnomethodology* [2]. The former consists of long-term observational studies of people in different contexts, ranging from anthropologists observing indigenous peoples in the bush to observing white collar professionals at work. Researchers attempt to describe behavior, seeking to identify general characteristics of human behavior as well as specific incidents of unique behavior. One of the roots of the word "ethnography" is "graph", which means "to write". Ethnographers, as scientists, are expected to contribute to a constantly-growing body of research literature, in which they compare and contrast their findings with those of other researchers.

Interactive system designers may profitably borrow observational techniques from ethnography, because they provide useful ways of observing and interpreting behavior in real-world contexts. However, the purpose is quite different. The designer uses ethnomethodology, i.e. methods from ethnography, to contribute understanding that is specific to the development of a particular software system. As with Psychology experiments, the particular techniques may be very similar but the context and underlying assumptions are quite different.

Engineering poses a different set of problems. One concern is that engineers are usually trained to solve problems that have been given to them and are evaluated on the technical validity of their solution, not the relevance of the problem. Yet designing a system by strictly following a set of design requirements does not guarantee a successful product. Human users add complexity and unpredictability to the situation and solutions that appear correct on paper may not be valid in practice. Software engineers are not taught strategies for questioning the design problem, so they often find themselves solving the wrong problems and ultimately failing to meet the needs of their users. Creating formal models of users and simulations of their activities provides a comforting feeling of having considered user's needs, until the software is actually used. Technical expertise is essential to the development of quality interactive software, but that technical expertise must be used to software the "right" problems.

The design disciplines, such as graphic design and architecture, represent the third critical component of interactive system design. Unlike engineers, designers *are* trained to question the 'design brief" and come up with alternative solutions. They have a very hands-on, apprentice-based learning process, in which they create designs for their

portfolios, which are critiqued by faculty and fellow students. However, in many design schools, the needs of the user are not reflected in the design brief, or if they are, designers are given few tools to actually determine those needs. Designers must develop their own methods for finding out about users and are not taught strategies for objectively comparing design decisions.

Each discipline offers valuable skills and perspectives; each has the potential to miss important aspects of the design problem. Multi-disciplinary design teams offer a solution, covering the full spectrum of design approaches, taking advantage of the strengths offered by each discipline while mitigating potential blind spots. However such teams pose another problem: participants must be able to communicate effectively with each other. The next section describes some of the issues designers face when attempting to work in a multi-disciplinary design team.

Working in Multi-disciplinary Design Teams

In the previous section, I identified some of the characteristics of the disciplines that provide fundamental contributions to interactive system design. Each have long-standing academic and professional traditions, with different values and specific research or development techniques. When someone trained in one of these "traditional" disciplines begins to work on the design of interactive software, he or she is faced with a problem: how to reconcile the differences between what was learned and how it is applied in the new design context. Most social scientists aren't taught the differences between research studies in a scientific and a software design context: they must discover this on their own. Similarly, engineers often discover that the design requirements are a moving target and they have not been given strategies for successfully developing code in such a dynamic environment. Designers may also be frustrated, since their work is suddenly subject to different kinds of critiques and evaluation than they faced in design school.

As educators, we face the question of how to train people to become successful interaction designers. One strategy might be to try to develop expertise in all of the component disciplines, teaching scientific, engineering and design principles. However, it is unlikely that many individuals will become expert in everything: it is far more likely that individuals will show talent in one area. A gifted artist may be enjoy drawing

and design but may find systematic observation of users or programming software to be difficult or uninteresting. Similarly, a trained observer of people may be able to contribute greatly to the understanding of the user's work, but may not be able to program or create elegant interface designs. A talented programmer may find talking to users or brainstorming interface design ideas equally difficult. So, while a few talented people may be able to contribute effectively in all areas, it is far more likely that they will find themselves contributing their expertise as part of a multi-disciplinary design team.

We have a different strategy, which is to continue training people from within their chosen major disciplines, whether scientific, engineering or design, but to increase their understanding and appreciation for the other disciplines. Students are exposed to different value systems and discuss how they may interact with each other.

Although ensuring that each person understands the perspectives of the others is important, it is rarely sufficient. We have found it necessary to create design activities in which all members of the design team, including users, can participate equally. These design techniques are borrowed from the full range of sub-disciplines and we discuss with students the implications of using them in a design, rather than their original, context. We choose techniques that increase communication among participants and we encourage students to develop new techniques that cross disciplinary boundaries. The next section describes some of these techniques, borrowed or inspired from various component disciplines described above.

Hands-on Interactive Design Techniques

Interaction design is an iterative process, as illustrated in figure 2. Students, whether at the University level or professionals in the field, are expected to participate in all of the design activities, throughout the design process. Although the process is presented as circular, it is important to recognise that, once begun, the design team can and should revert to any of the earlier stages as necessary.

We begin by "finding out about users", using techniques drawn from the social sciences and design. We then work on generating a design space and expanding it by creating new ideas. Once we have a suitably rich design space, we begin to select particular

design directions and begin prototyping a design. At any point, we may decide that we need additional information about users or new ideas to help make design choices. At various points through the development, we evaluate our the design, beginning with early prototypes and continuing through to the final working system.

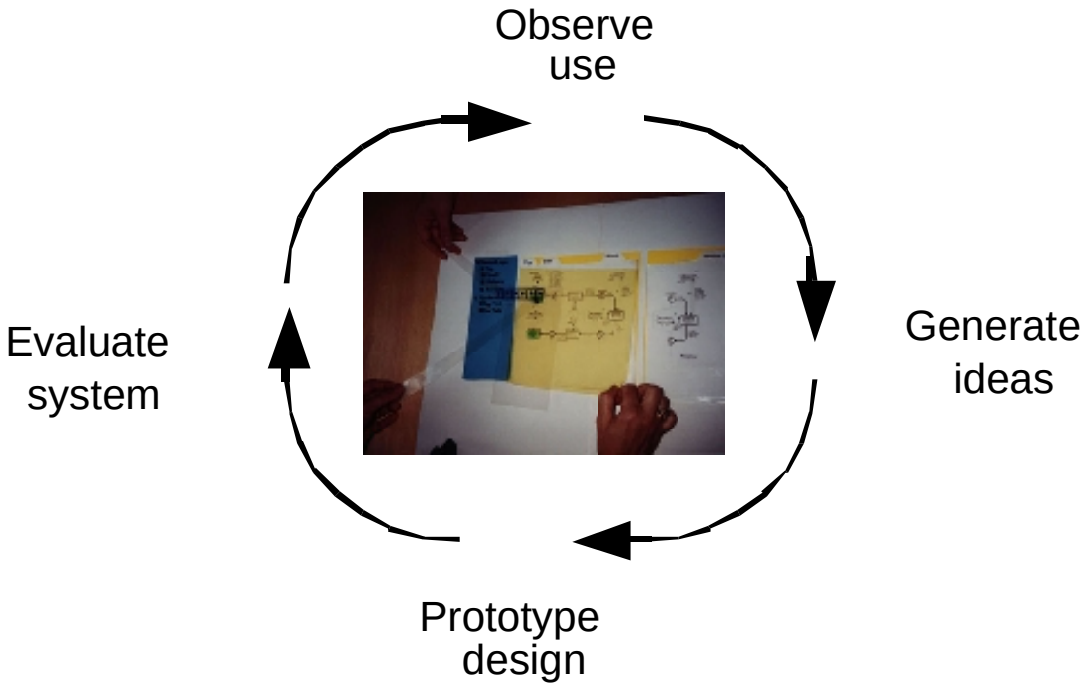

Figure2: The interaction design process is highly iterative and requires techniques for finding out about users, generating new ideas, developing design prototypes and evaluating aspects of the system.

Table 1 summarizes a set of observation and design techniques that we have adapted from various disciplines or have developed explicitly. We believe in the concept of "triangulation" [10, 12], in which we use multiple design methods to help us avoid particular design biases. We use these techniques in our own research and development work, as well as for teaching: these are the techniques that have stood the test of time. They are simple to use, speed rather than hinder the design process, and all serve to increase communication within the design team and among designers, users and various other stakeholders.

(Note: The video-based techniques in table 1 are illustrated in a DVD tutorial by Mackay [13], available through ACM/SIGCHI.)

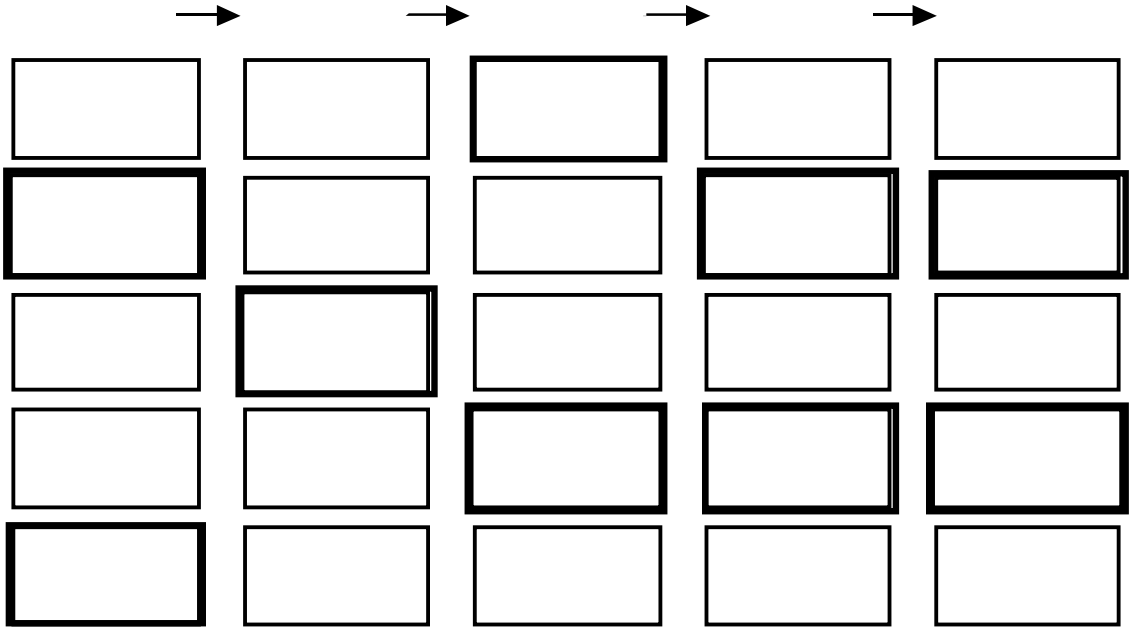

Table 1: Design techniques derive from a variety of disciplines.
Items with bold outlines are described in further detail in the text.

Finding out about users

The first phase of the development process involves finding out about the needs and desires of the future users of the system being designed. Sometimes a system already exists, sometimes not. In any case, it is important to find out about the context in which the system will be use in order to begin to define the design problem.

From the social sciences, we use observation and interview techniques [15]. Video is a useful tool, although it should be used judiciously: I tell my students to only videotape what they are willing to later watch. With respect to interviews, we have borrowed a technique introduced by Flanagan [7] from the human factors community, called "critical incident technique". This and other related techniques are based on an important observation about people: if you ask people specific questions they will give you specific answers. You can then generalize or ask them to generalize for you. If you instead begin with general questions (such as "How do you use your email?") you will receive general answers that provide very little that is useful for designing a future system. So, when interviewing people, the key is to find specific objects, events or times that people can describe and elaborate on. This specific information that can then be woven into design scenarios.

From design we have the notion of cultural probes [6] from Gaver and his colleagues. Here, the emphasis is not on collecting data but rather on involving future users and helping them generate inspiration for design. Cultural probes are specific objects, such as a map to create or a camera to take photographs, that users use to comment on their existing world and to generate ideas about future possibilities.

We have recently been experimenting with a new method that we call technology probes[8] which attempt to incorporate both scientific data collection and design inspiration. For example, for our interLiving Disappearing Computer project, which studies technologies for distributed families, we have created simple, limited-functionality prototypes that we have placed in family members' homes. These probes provide direct, private links between households and enable sharing of video images or hand-written messages. Technology probes designed to both collect information, informing us about the communication patterns within the families, and to provoke new ideas, inspiring both the family members and us as designers to create new technologies to meet needs we had not previously observed.

Once we collect information about users, we need to analyse or interpret it, not for its own sake but to inform design. It is important to preserve the context of the user's activities: we do not try to abstract out a set of abstract tasks, but rather seek to present each user's activities in context. The most effective strategy we have found is to develop scenarios [3, 9], which combine the experiences, both typical and unusual, of different real users. We begin by creating a "day-in-the-life" story, and then break the story up into an illustrated storyboard. Sometimes, we create video scenarios, either with video clips from actual observations or re-enactments of events we've observed. These scenarios provide an effective communication tool for all members of the design team, and give us a way to discuss what we've learned with the users [16], who can give us feedback and enrich the scenarios.

Creating a design space

The second phase of the development process involves the creation of a design space [1]. Here, the goal is to generate new ideas and to increase the set of design possibilities. Brainstorming [4] is the most common technique: The classic procedure involves a small number of people who are given a specific topic and a limited period of time.

Everyone participates in generating ideas, all of which are captured on a blackboard or flip chart. Another variation asks everyone write down ideas individually and then share them with the group. A moderator ensures that all comments are constructive, that the time is spent generating ideas, not evaluating them, and that the session finishes on time. The time limit is very important: brainstorming is very intense and, if done well, will leave everyone energized and excited by the ideas, not tired and bored. Brainstorming usually has two phases: the first for generating ideas and the second for reflecting upon them.

We have discovered [14] that the quality of the ideas change according to the way they are created. Verbally shouting out ideas, as recommended in classic brainstorming, is effective for rapidly generating large quantities of ideas, but the ideas themselves are poorly developed and often vague. People quickly lose the context in which the ideas were created and flipcharts from month-old brainstorming sessions are mostly useless. Drawing ideas rather than saying them requires more reflection and other participants often have an easier time understanding them. We push this further, by asking participants to show their ideas, via paper or more elaborate prototypes. This forces both idea-generators and other participates to concentrate on what it will be like for users to interact with the idea in question. Such ideas become more concrete and we find that they are more likely to inspire further ideas.

For us, the most effective technique is video brainstorming [1] in which participants demonstrate their ideas in front of a video camera, using rapid paper or other prototypes. Not only does this produce a more valuable record of each idea, which can be reviewed and expanded upon later, but it is very effective for encouraging participants to think concretely about how users will actually interact with the proposed idea. Video brainstorming also forces active participation from everyone. Each idea has an author, who directs other members of the group to play the role of the user or the system to illustrate the interaction. Video brainstormed ideas allow participants to "sketch" interaction ideas and share them, even if they are not expert programmers or graphic artists. We have handed video brainstormed ideas to programmers, who can rapidly prototype code and allow everyone to explore the ideas further. We also find this an excellent technique for working with users, who can contribute directly to the design process without any particular technical skills. Once the team is used to it, video brainstorming is only slightly more time-intensive than other forms of brainstorming,

but we find it much more useful, since the resulting video record of design ideas continues to serve as a source of inspiration throughout the design process.

Prototyping a design

The third phase of involves making choices: deciding to pursue some directions and omit others [1]. Unlike the idea generation phase, which values quantity not quality of ideas, the purpose of this phase is to explicitly narrow the range of possibilities and choose a particular path. The goal is to explore a more restricted design space, considering the details of each design decision and creating a grounded design that is both innovative and still makes sense to real users in real-world contexts.

We use a variety of prototyping techniques, ranging from very rapid, paper prototypes to intermediate software prototypes, from "Wizard of Oz" [11] techniques to working systems. When we develop video prototypes, we revise the use scenarios that we created in the first phase of the design process and explore how a new design would be used in that context. We develop the system design and the scenario together, changing each to meet the needs of the other. Once we have several scenarios that illustrate the use of the new design, we create storyboards and prototyping materials, and illustrate the design ideas with a video prototype.

This process is very effective for giving all participants in the design team, especially users, a voice in the process. Everyone can see what the design implications are for particular design decisions, and everyone can suggest and show alternatives. If people disagree, they can return to techniques from the earlier design phases to gather more information or generate alternative design ideas.

Evaluating a system

The final phase involves evaluating the design: is it a successful solution? Are there specific problems that need to be fixed? Do the users like it? We run various kinds of studies to answer such questions. Sometimes, it is important to run controlled experiments. However, usually, it is more important to simply find a number of users and watch them use the new system. We usually ask pairs of users to sit together and comment on the system out loud, which makes it easier for them to express their

opinions to us. Normally, we ask them to try a set of tasks or run through several scenarios, and simply watch how well they are able to use the system. In addition to videotaping them, we use the computer to log their interaction with the software, so we can obtain quantitative data about errors and efficiency of different user actions.

Another useful strategy is a design walkthrough, based on Yourdan's [17] work with structured walkthroughs. A "walkthrough" is a peer group review of a product: people at roughly the same level in the organization meet to systematically review and discuss a segment of software. One can review code, architecture or any aspect of the software, including video prototypes. The rules are simple, but important: Groups should be small (3-7 people), members of the group should be at the same level, the presenter should prepare in advance, everyone must be on time and the review should be limited to at most one hour. The goal of the walkthrough is to identify as many problems as possible, not to discuss solutions. Criticisms should be as positive as possible and should be restricted to the design at hand. Walkthroughs are similar in format to brainstorming sessions, but opposite in their goals: walkthroughs seek to find problems, brainstorming sessions seek to maximize the number of ideas.

Conclusion

This paper has described our strategy for teaching interaction design. We begin with the recognition that design is multi-disciplinary and that few individuals can be expert in all of the necessary fields. We teach our students how to think about the design perspectives of their colleagues: what are the most important contributions of each design field and what are the potential sources of misunderstanding? We also teach specific, hands-on design techniques that draw from all of the component disciplines of human-computer interaction. The techniques described in this paper are explicitly intended to equalize the level of the participants, enabling everyone to actively contribute, including users. Using these strategies not only improves communication among members of the design team (and users!) but also improves the efficiency and effectiveness of the design process. People can explore a wider range of ideas, and select promising design solutions earlier, with greater relevance to users, using these design techniques. Of course, team members with specific skills in specific domains, such as interviewing, programming or graphic design, will not only be able to contribute

their expertise, but will also benefit from knowing that others will recognize the value of their contributions. Finally, these techniques are fun; participants of multi-disciplinary design teams should enjoy designing interactive systems!

References:

1. Beaudouin-Lafon, M. and Mackay, W.E. Prototyping Development and Tools. In J.A. Jacko and A. Sears (Eds), *Handbook of Human-Computer Interaction*. New York: Lawrence Erlbaum Associates (60 pages), 2002

2. Button, G. and Dourish, P. Technomethodology: Paradoxes and Possibilities. *Proceedings of the CHI '96 conference companion on Human factors in computing systems*, pp. 19 - 26 1996

3. Carroll, J. *Scenario-Based Design: Envisioning Work and Technology in System Development.* NY: Wiley, 1995

4. Clark, C. *Brainstorming : How to Create Successful Ideas.* CA: Wilshire Book Company, 1989

5. Djkstra-Erikson, E., Mackay, W.E. and Arnowitz, J. Trialogue on *Design of. ACM/Interactions*, pp. 109-117, March, 2001

6. Gaver, W. and Dunne, A. Projected Realities, Conceptual Design for Cultural effect. *Proceedings of ACM Conference on Human Factors in Computing Systems CHI '99*, p. 600-607, 1999

7. Flanagan, J. The Critical Incident Technique. *Psychological Bulletin.* 51(4). pp. 327-358, 1954

8. H. Hutchinson, W. Mackay, B. Westerlund, B.B. Bederson, A. Druin, C. Plaisant, M. Beaudouin-Lafon, S. Conversy, H. Evans, H. Hansen, N., Roussel, B. Eiderbäck, S. Lindquist, Y. Sundblad Technology Probes: Inspiring Design for and with Families, *Proceedings of ACM Conference on Human Factors in Computing Systems, CHI 2003*, Fort Lauderdale (USA), April 2003, CHI Letters 5(1), ACM Press, 2003

9. Mackay, W. & Bødker, S. Workshop on Scenario-Based Design. In *CHI'94 Conference Companion.*, Boston, MA: ACM Press, 1994.

10. Mackay, W.E. and Fayard, A-L. HCI, Natural Science and Design: A Framework for Triangulation Across Disciplines. *Proceedings of ACM DIS '97, Designing Interactive Systems*. Amsterdam, Pays-Bas: ACM/SIGCHI, pp.223-234, 1997

11. Mackay, W.E. Beyond the Wizard of Oz. *CHI '86 Conference on Human Factors in Computing Systems.* Boston, MA: ACM/SIGCHI, 1986

12. Mackay, W.E. Triangulation within and across HCI disciplines. *Human-Computer Interaction*. Hillsdale, New Jersey:Lawrence Erlbaum Associates. Invited Commentary on the article: "Damaged Merchandise? A Review of Experiments that Compare Usability Evaluation Methods", W.D Gray and M.C. Salzman. Vol. 13, #3, pp. 310-315, 1998

13. Mackay, W.E. Using Video to Support Interaction Design. DVD Tutorial, *CHI'02 Conference on Human Factors in Computing Systems,* Minneapolis, MN ACM/SIGCHI. 2002

14. Mackay, W.E., Ratzer, A., and Janecek, P. Video artifacts for design: Bridging the gap between abstraction and detail. *Proceedings of ACM DIS 2000, Conference on Designing Interactive Systems.* Brooklyn, New York. ACM Press. pp. 72-82, 2000

15. Patton, M.Q. Qualitative Interviewing. In *Qualitative Evaluation and Research Methods*, Sage Publications, pp. 227-359, 1990

16. Westerlund, B., Lindqvist, S., Mackay, W., and Sundblad, Y. Co-design methods for designing with and for families. *Proceedings of EAD'03, the fifth European Academy of Design conference*, Barcelona, Spain, 2003

17. Yourdan, E. *Structured Walkthroughs.* NY: Prentice-Hall, 1979