# Exploring Richer Ecologies between Designers and Users

**Gerhard Fischer**

University of Colorado at Boulder
gerhard@colorado.edu

**Abstract** The fundamental contribution of Semiotic Engineering to human-computer interaction has been to propose and establish a discourse to conceptualize HCI not primarily about how users interact with computers but as computer-mediated communication between designers and users at interaction time. It represents an important contribution to explore the future of HCI beyond the focus of usability and interfaces. In our own research over the last few decades, we have also explored alternative approaches to create richer frameworks and ecologies how designers and users can interact in the design of computational artifacts. This contribution briefly describes some of these approaches by characterizing the problems that they addressed and the potential solutions that they explored.

## Introduction

Semiotic Engineering views HCI as "computer-mediated communication between designers and users at interaction time. The system speaks for its designers in various types of conversations specified at design time. These conversations communicate the designers' understanding of who the users are, what they know the users want or need to do, in which preferred ways, and why. The designers' message to users includes even the interactive language in which users will have to communicate back with the system in order to achieve their specific goals." ([deSouza & Leitao, 2009], p. vi) Reconceptualizing HCI not primarily as interactions between users and systems, but as communication processes between designers and users with the computational environments serving as media, emphasizes different design challenges, frameworks, and objectives.

Some of Clarisse Sieckenius de Souza's research and our research explored some closely related issues, e.g. to conceptualize software development [deSouza et al., 2016; Fischer & Nakakoji, 1992] as reflective practice [Schön, 1983], we have also explored alternative approaches to Semiotic Engineering to create richer frameworks and ecologies on how designers and users can interact in the design of computational artifacts. While some of these research activities (addressing problems, creating frameworks, and designing, developing, and assessing systems) have taken place some time ago, this brief contribution tries to argue that at least some aspects still represent challenges today complementing and enriching Semiotic Engineering approaches in exploring the future of HCI beyond usability and interfaces [Greenberg & Buxton, 2008].

## Advances and Reconceptualizations of HCI

A considerable amount of research has been conducted in HCI during the last 30 years. The early research efforts centered on complementing professionally dominated design with user-centered design [Norman & Draper, 1986] and they were driven by *technological* developments that line-oriented interfaces were replaced by WIMP interfaces and by *societal* developments that computer users were coming from all fields rather than being only computer scientists. An important vision (at the time) was provided by Newell and Card [Newell & Card, 1985] in the mid-eighties to conceptualize HCI at different time scales (see Fig. 1).

| | Time | Action | Memory | Theory |
|---|---|---|---|---|
| (sec) | (common units) | | | |
| $10^9$ | (decades) | Technology | Culture | Social and Organizational |
| $10^8$ | (years) | System | Development | |
| $10^7$ | (months) | Design | Education | |
| $10^6$ | (weeks) | Task | Education | |
| $10^5$ | (days) | Task | Skill | Bounded Rationality |
| $10^4$ | (hours) | Task | Skill | |
| $10^3$ | (ten mins) | Task | LTM | |
| $10^2$ | (minutes) | Task | LTM | |
| $10$ | (ten secs) | Unit task | LTM | Psychological |
| $1$ | (secs) | Operator | STM | |
| $10^{-1}$ | (tenths) | Cycle time | Buffers | |
| $10^{-2}$ | (centisecs) | Signal | Integration | Neural And Biochemical |
| $10^{-3}$ | (millisecs) | Pulse | Summation | |

Fig. 1 – Time Scales of Human Action ([Newell & Card, 1985], p. 226)

As the envisioned developments started to take place in reality, namely the time scales becoming longer (i.e.: weeks, months, and years rather than seconds or minutes), the research challenges were grounded in a world in which social theory became increasingly relevant and complemented the research activities focused on the isolated actor. These developments required that HCI research got more closely linked with theories and developments from education, development, and cultural change.

Our own research methodology was grounded in the philosophical approach of Popper focused on *"the search for knowledge does not start from perceptions, or observations, or collection of data or facts, but it starts from problems."* [Popper, 1959]. Table 1 provides an overview of the topics of our research that were driven by addressing problems that HCI research needed and needs to address.

Table 1: Overview of Themes

| **Theme** | **Problem** | **Potential Solution Explored** |
|---|---|---|
| "usable *versus* useful" to "usable *and* useful" | reality is not user-friendly requiring high-functionality environments | learning on demand, achieve external simplicity with internal complexity |
| bridging situation and system models | people (e.g.: designers and users) live in different conceptual environments | building a variety of different bridges to creates common ground and shared understadning |
| human problem-domain interaction | thin spread of application knowledge | domain-oriented design environments |
| meta-design | closed systems cannot models open and changing worlds | empowering users to act as designers; breaking down the strict separation between designers and users; democratizing innovation |

## From "Usable *versus* Useful" to "Usable *and* Useful"

HCI approaches have traditionally focused on making systems more *usable* — often by reducing the expressive power of the systems and their interfaces. This approach has focused on designing systems for casual users who are assumed to be using the system for the first time, for only few times, and for simple activities. Walk-up-and-use systems, such as ATMs (Automated Teller Machines), are examples of low-threshold, low-ceiling systems; they are be easy to understand and use without prior experience, but are limited in the power they afford users.

In contrast, systems for professional use need to be *useful*; these systems are used daily for a wide range of activities within their application domains, such as information gathering, representing design situations, exploring alternative and sharing work with colleagues. The needs of professionals have traditionally been addressed with high-threshold and high-ceiling systems that can be difficult to use at first, but over time are learned and support users to perform a wide range of tasks.

Designing useful *and* usable high-functionality environments (HFEs) [Fischer, 1993] requires that we go beyond the traditional HCI notions of "usability" and "ease of use," to design and assess HFEs that interact with designers, individually and collaboratively, as they struggle with large and ill-defined problems over long periods of time.

HFEs are a result that "reality is not user friendly" — therefore many tools are required [Buxton, 2002]. Users can be overwhelmed by having to decide which way is best, rather than merely finding a single way to do the task. In many cases, HFEs

- create a "*tool-mastery burden"* requiring users to spend considerable effort to master the system before they can do anything useful with it thereby outweighing the advantage of the broad functionality offered;
- most of their functionality goes unused, and users need support to find the tools or information they need when they need it; and
- no system, regardless of its size and complexity can achieve complete coverage of large and changing domains. Inevitably, some information will become obsolete, and breakdowns due to missing knowledge will occur.

To make systems useful *and* usable HFEs needed to be improved in their ability to interact with users by giving them "knowledge" about the problem domain, about communication processes, and about the users who interacted with the system. Our approaches to achieve these objectives have been:

- to address the "tool-mastery burden" with domain-oriented systems that allow users to communicate with the problem domain rather than the computer. Human problem-domain interaction allows users to spend more of their time thinking about their tasks, and less time worrying about communicating with the computer;
- to address the "unknown functionality" challenge by developing techniques to give systems the ability to determine what information and functionality might be relevant to the tasks a user is trying to accomplish, and to actively present users with this information without having them to explicitly ask for it; and
- to address the "impossibility of complete coverage" challenge by viewing systems as open-ended and continuously adapted by the people who use them in their day-to-day work.

## Bridging Situation and System Models

The interaction between users and systems is a conversation in a vocabulary and language determined by the input the system is able to accept and process. In many systems, information is structured around a description of the system, not around an analysis of the problems users address when using the system.

Compared to humans, computers are able to respond to a very limited range of input [Suchman, 1987]. To get something done on a computer system, users must provide input to the system that is within the limited range to which the computer is programmed to respond. To talk about the discrepancies between the way a human thinks of a problem and the limited inputs to which the system is capable of responding, we have used the terms *situation* and *system model* [Kintsch, 1998]:

- the *situation model* is a mental representation of the situation as the user sees it, including the problems motivating a task, general ideas for finding a solution, and a characterization of the desired goal state;
- the *system model* consists of a set of operations that, when invoked, will result in the desired solution. These operations must all be within the repertory of the system: for each operation there must exist one or more commands, depending upon context, to execute it.

At the level of the situation model, goals refer to actions and states in the users' problem space and are articulated in terms of what they want. Goals may be precise or imprecise, but the important point is that they are structured or named according to the system model [Furnas et al., 1987] rather than in terms of the application problem to be solved [Curtis et al., 1988]. Fig. 2 illustrates several different approaches to dealing with the gap between situation and system models:

- In (1) there is no support for bridging this gap. In this case, people frequently have difficulties in solving problems or finding information, because they are unable to communicate within the scope of the system model, even if they have a clear understanding of what they want to do.
- In (2), a new system model is constructed which is closer to an individual's situation model and hence more intuitive and easier to use representing the objective of human problem-domain interaction.
- In (3), users are supported to bring their situation model closer to the system model by making the relevant features of the latter more transparent. An example of this approach is the paradigm of retrieval by reformulation [Williams, 1984]. This theory postulates that people naturally think about categories of things not in terms of formal attributes but in terms of examples. Through presenting examples of objects retrieved by queries, the system reveals its system model and helps the user to incrementally formalize their query to better match the system model.
- In (4), a knowledge-based agent translates a request in the situation model into the system model requiring the agent to have enough knowledge to assist users in mapping tasks conceptualized in their situation model to the system model.
- In (5), users are trained to express themselves in the system model. This is the case with many skilled expert users, who have gained over time a familiarity with the system model such that their situation model matches the system model.

| Different Architectures and Support | Situation Model | System Model |
| --- | --- | --- |
| (1): no support for bridging the gap between situation and system models | | |
| (2): system models are constructed closer to individuals' situation models | | |
| (3): bring their situation model closer to the system model by making the relevant features of the latter more transparent | | |
| (4): knowledge-based agents translates requests in the situation model into the system model | | |
| (5): users are trained to express themselves in the system model | | |

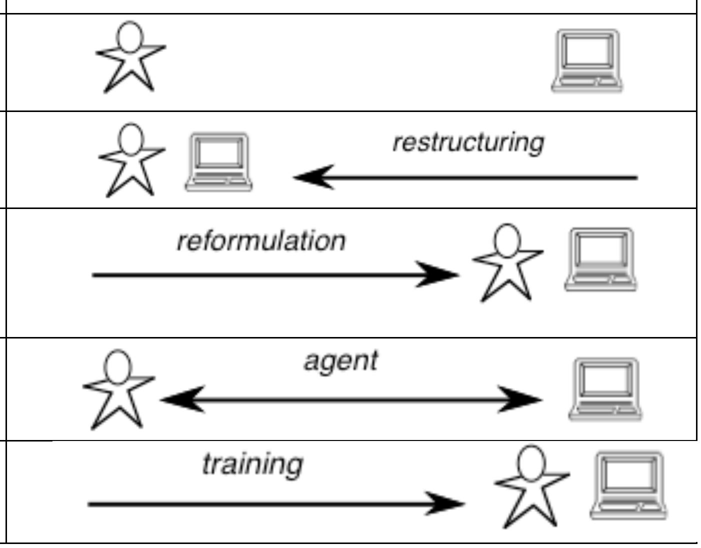


Fig. 2 – Different Relationships between Situation and System Models

## Human Problem-Domain Interaction

If the most important role for computation is to provide people with a powerful medium for expression, then the medium should support them in working on the task, rather than require them to focus their intellectual resources on the medium itself. When users suffer from a tool mastery burden, their tasks fade to the background while effort is put toward mastering the tool. To bring tasks to the forefront, computers must become "*invisible*" [Norman, 1991] allowing users to put the majority of their efforts toward interacting with the problem domain, rather than the computer.

The shift from "human computer interaction" to "human problem-domain interaction" [Fischer & Lemke, 1988] puts users, being the owners of problems, in charge freeing them from a reliance on computer specialists. In analogy to eliminating the power of scribes in the middle ages (when most people were unable to read and write), a major goal of HCI should be "to take the control of computational media out of the hands of high-tech scribes."

Professional programmers and end-users define the endpoints of a continuum of computer users. The former like computers because they can program, and the latter because they get their work done. Techniques and languages enabling users to acquire computer literacy have been explored with *end-user development* [Lieberman et al., 2006; Nardi, 1993]. One objective of *end-user development* is to offer task-specific languages that take advantage of existing user knowledge (e.g., mathematicians already know the mathematical knowledge embedded in Mathematica, and accountants already know the conceptual model behind spreadsheets [Fischer & Rathke, 1988]).

## Meta-Design: Empowering Users to Act As Designers

*Meta-design* [Fischer & Giaccardi, 2006] represents a theoretical framework, supported by innovative information and communication technologies, in which humans of all ages can participate as co-designers and pursue topics of interest, reformulate knowledge, express themselves creatively and appropriately, and produce and generate information rather than simply comprehend existing information.

In a world that is not predictable, improvisation, evolution, and innovation are more than luxuries — they are *necessities*. The challenge of design is not a matter of getting rid of the emergent, but rather of including it and making it an opportunity for more creative and more adequate solutions to problems. Many design approaches force all the design intelligence to the earliest part of the design process, when everyone knows the least about what is really needed.

Meta-design provides the foundations for *cultures of participation* [Fischer, 2011] in which all users can act as active contributors at use time. In cultures of participation, users are allowed and empowered to make significant modifications when the need arises because, despite the best efforts at design time, systems need to be evolvable to fit new needs and account for changing tasks. Meta-design complements and transcends *participatory design* [Binder et al., 2011] that seeks to involve users more deeply in the process as co-designers by empowering them to propose and generate design alternatives themselves. Participatory design requires the social inclusion and active participation of the users at *design time* by bringing developers and users together to envision the contexts of use. Meta-design provided a foundation and its was further developed in long-term research project "The Envisionment and Discovery Collaboratory" [Arias et al., 2016], a table-top computing environment in which all stakeholders could participants as active contributors in personally meaningful problems.

## Broad Impact

This contribution contrasted semiotic engineering with some alternative conceptual developments that we have pursued in our own research activities. There are numerous other frameworks developed that are relevant for the future of HCI. Three representative developments that I consider personally important to take into account are will be briefly mentioned here: creativity support tools, social production, and libertarian paternalism.

**Creativity support tools** [Shneiderman, 2007] enable new forms of expression for individuals, groups and communities. Their focus is to move HCI beyond usability, usefulness, and productivity objectives to support creativity and innovation and to form new alliances between research communities in the creative practices and information and communication technologies.

**Social production** [Benkler, 2006] provides the foundation that social media enable many more individuals to communicate their observations and their viewpoints to many others, thereby changing control and fostering new levels creativity in digital cultures. Social production emphasizes that technology alone does not determine social structure: it creates feasibility spaces for new social and cultural practice. Meta-design supports social production by facilitating and supporting the creation of consumer-generated content and it enriches individual autonomy substantively by creating an environment built less around control and more around facilitating action.

**Libertarian Paternalism** [Thaler & Sunstein, 2009] represents a fundamental concept applicable in many domains that it is both possible and legitimate for private and public institutions to affect behavior while also respecting freedom of choice of individuals belonging to these institutions. It argues that designers should consider themselves as choice architects thereby promoting a design methodology. The framework demonstrates that designer-imposed structures affecting users' choices are inevitable, and hence that a form of paternalism cannot be avoided but that the libertarian part allows users to override the choices made by designers.

## Conclusions

Semiotic engineering has provided insights and theoretical foundations for HCI in the past and will continue to do so in the future. Beyond its impact on HCI, it has influenced many scientific fields including computer-mediated communication, multimedia-hypermedia systems, end user development, intelligent user interfaces, artificial intelligence, and various branches of software engineering. My hope is that the ideas and developments briefly discussed in this contribution can form an alliance with semiotic engineering to create elements for innovative frameworks in HCI that will make a difference with respect to the problems that we frame and solve, the questions that we ask, and the research activities that we undertake.

## References


Arias, E. G., Eden, H., & Fischer, G. (2016) *The Envisionment and Discovery Collaboratory (EDC): Explorations in Human-Centered Informatics,* Morgan & Claypool Publishers, San Rafael, California.

Benkler, Y. (2006) *The Wealth of Networks: How Social Production Transforms Markets and Freedom,* Yale University Press, New Haven.

Binder, T., DeMichelis, G., Ehn, P., Jacucci, G., Linde, P., & Wagner, I. (2011) *Design Things,* MIT Press, Cambridge, MA.

Buxton, W. (2002) "Less Is More (More or Less)" in P. J. Denning (Ed.), *The Invisible Future - the Seamless Integration of Technology in Everyday Life,* McGraw-Hill, New York, pp. 145-179.

Curtis, B., Krasner, H., & Iscoe, N. (1988) "A Field Study of the Software Design Process for Large Systems," *Communications of the ACM,* 31(11), pp. 1268-1287.

deSouza, C. S., Cerqueira, R. F. d. G., Afonso, L. M., Brandão, R. R. d. M., & Ferreira, J. S. J. (2016) *Software Developers as Users: Semiotic Investigations in Human-Centered Software Development,* Springer International Publishing.

deSouza, C. S. & Leitao, C. F. (2009) *Semiotic Enginnering Methods for Scientific Research in Hci,* Morgan & Claypool Publishers, San Rafael, California.

Fischer, G. (1993) "Beyond Human Computer Interaction: Designing Useful and Usable Computational Environments" in *People and Computers Viii: Proceedings of the Hci'93 Conference (Loughborough, England),* Cambridge University Press, Cambridge, UK, pp. 17-31.

Fischer, G. (2011) "Understanding, Fostering, and Supporting Cultures of Participation," *ACM Interactions* XVIII.3 (May + June 2011), pp. 42-53.

Fischer, G. & Giaccardi, E. (2006) "Meta-Design: A Framework for the Future of End User Development" in H. Lieberman, F. Paternò, & V. Wulf (Eds.), *End User Development,* Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 427-457.

Fischer, G. & Lemke, A. C. (1988) "Construction Kits and Design Environments: Steps toward Human Problem-Domain Communication," *Human-Computer Interaction,* 3(3), pp. 179-222.

Fischer, G. & Nakakoji, K. (1992) "Beyond the Macho Approach of Artificial Intelligence: Empower Human Designers - Do Not Replace Them," *Knowledge-Based Systems Journal, Special Issue on AI in Design,* 5(1), pp. 15-30.

Greenberg, S. & Buxton, B. (2008) "Usability Evaluation Considered Harmful (Some of the Time)" in *CHI 2008 Proceedings,* ACM, Florence, Italy, pp. 111- 120.

Kintsch, W. (1998) *Comprehension: A Paradigm for Cognition,* Cambridge University Press, Cambridge, UK.

Lieberman, H., Paterno, F., & Wulf, V. (Eds.) (2006) *End User Development* Kluwer Publishers, Dordrecht, The Netherlands.

Nardi, B. A. (1993) *A Small Matter of Programming,* The MIT Press, Cambridge, MA.

Newell, A. & Card, S. K. (1985) "The Prospects for Psychological Science in Human-Computer Interaction," *Human-Computer Interaction,* 1(3), pp. 209-242.

Norman, D. (1991) "Cognitive Artifacts" in J. M. Carroll (Ed.), *Designing Interaction,* Cambridge University Press, Cambridge, pp. 17-38.

Norman, D. A. & Draper, S. W. (Eds.) (1986) *User-Centered System Design, New Perspectives on Human-Computer Interaction,* Lawrence Erlbaum Associates, Inc., Hillsdale, NJ.

Popper, K. R. (1959) *The Logic of Scientific Discovery,* Basic Books, New York.

Schön, D. A. (1983) *The Reflective Practitioner: How Professionals Think in Action,* Basic Books, New York.

Shneiderman, B. (2007) "Creativity Support Tools: Accelerating Discovery and Innovation," *Communications of the ACM,* 50(12), pp. 20-32.

Suchman, L. A. (1987) *Plans and Situated Actions,* Cambridge University Press, Cambridge, UK.

Thaler, R. H. & Sunstein, C. R. (2009) *Nudge — Improving Decisions About Health, Wealth, an Happiness,* Penguin Books, London.

Williams, M. D. (1984) "What Makes Rabbit Run?" in *International Journal of Man-Machine Studies, Volume 21*, pp. 333-352.