Designing for Collaboration: Improving Usability of Complex Software Systems

Mari-Klara Oja
Bentley University
175 Forest St.
Waltham, MA 02452 USA
oja_mari@bentley.edu

CHI 2010, April 10–15, 2010, Atlanta, Georgia, USA.
ACM 978-1-60558-930-5/10/04.

Abstract
Designing for collaboration approaches systems and users as a team and focuses on the cooperation between the two. This work in progress aims to delineate how designing for collaboration is also inherently designing for usability. It is proposed that designing for collaboration is theoretically more appropriate for building complex problem-solving applications, where the user and system are by definition co-information-processors.

Keywords
Human-computer collaboration, usability, complex software systems

ACM Classification Keywords
H.5.2 User Interfaces: Theory and methods

General Terms
Theory

Introduction
In 1960, Licklider [8] coined the term man-computer symbiosis and wrote, "The hope is that, in not too many years, human brains and computing machines will be coupled very tightly, and that the resulting

partnership will think as no human brain has ever thought and process data in a way not approached by the information-handling machines we know today." In more recent years, a paradigm shift from human-computer interaction (HCI) to human-computer collaboration (HCC) has been proposed [2,4,12,13,15]. When computer information systems are used as aids in problem solving, users and systems become increasingly "co-information-processors," highlighting the need for "cognition friendly" information systems [2]. Grosz [4] points out that instead of designing "screen deep" human-computer interfaces that let the users talk to machines through interfaces, we should design collaborative interfaces that allow systems to work together with the user. Similarly, Shieber [13] argues that the goal of an interface is to allow users and computers to collaborate on problem solving and distribute tasks according to participant strengths. According to Terveen [15], a unified approach to creating collaborative systems entails both making computers more human-like and also exploiting their unique abilities to complement humans. Rich and colleagues [12] apply collaborative discourse theory to human-computer interaction to build intelligent collaborative software systems. Their approach focuses on mimicking human-human collaboration in human-computer collaboration.

The aim of this work in progress is to show how theoretically designing for usability is subsumed by designing for effective collaboration. Designing for collaboration takes a wider perspective by focusing on successful human-system cooperation, thus incorporating both the static elements of the interface as well as the dynamics of interaction. Specifically, the goal of this paper is to develop an initial semantic relation between usability and human-computer collaboration. Its main contribution is the preliminary mapping between usability and collaborativeness. To the author's knowledge, so far no such theoretical link has been systematically proposed.

Complex Software Systems

The paradigm shift from HCI to HCC is especially important when it comes to complex software applications – "systems, which are used to help structure and solve ill-structured problems" [5]. Decision support systems and safety-critical systems are often cited as examples of complex software applications [3,5]. Mirel [9] states that complex problem-solving involves "ill-defined situations; vague or broad goals; large volumes of data from many sources unprocessed for immediate purposes; nonlinear, often uncharted analytical paths; no pre-set entry or stopping points; many contending legitimate options; collaborators with different priorities; 'good enough' solutions with no one right answer; and underlying patterns structuring open-ended investigations that, due to contextual conditions and constraints, are never performed the same way twice". Building applications for complex problem-solving requires a different type of design process and methodologies, which Mirel calls designing for usefulness, where the goal of the application is not just to simplify work but to be operationally simple, while intellectually sophisticated and nuanced [9].

According to Haynes and Kannampallil [5], complex software applications require great cognitive skill, integration of knowledge from various areas, and advanced instruction and learning; thus, it is not surprising that "screen deep" interfaces to such

systems may not yield the best results in terms of usability. As Mirel [9] points out, most current HCI practices concentrate on ease of use or simplifying the work, and this may lead to "producing good designs but for the wrong problems." Focusing on the superficial elements (such as colors, size, labels, etc.) in usability improvements will probably result in a more intelligible interface, but when the goal is to help users solve complex problems, this may not be enough. In such cases, it is essential that users and systems understand their respective roles and how those fit into the overall task.

Other design and engineering methods, besides designing for usefulness, have been proposed to deal with complex software systems. Cognitive engineering [3] and learner-centered design [14] focus on improving system-human cognitive fit and allowing users to construct better mental models (knowledge) of the system. This work builds on and adds to these concepts by suggesting that, by shifting our view from complex software systems as tools to complex software systems as effective collaborators, we can build better and more usable systems without losing the necessary complexity.

Usability

Usability is commonly defined as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [6]. According to Nielsen [10], usability is mainly about how easy an interface is to use, and it consists of five quality attributes – learnability, efficiency, memorability, errors and satisfaction. Although, in recent years, the concept of user experience [7] has become popular and new definitions, models and methods are proposed frequently, the standard view of usability is still central to HCI. This paper adopts a wide notion of usability by understanding effectiveness, efficiency and satisfaction to include also such things as usefulness, joy of use and the general user experience. The paper proposes that designing for collaboration implies designing for such wider usability.

Human-Computer Collaboration

Human-Computer Collaboration is a process in which human(s) and computer system(s) work together and coordinate their actions to achieve shared goals [12,15]. Terveen [15] proposes that human-computer collaboration should incorporate what he calls the "human emulation" and the "human complementary" methods. Thus, designing for collaborative systems includes both mimicking and modeling human-human collaboration and also the strategic division of tasks according to the asymmetric abilities of computers and humans.

Terveen [15] brings out four characteristics of this unified approach: reification, balance between representation/reasoning and interaction, natural communication, and collaborative adaptation. An inherent strength of graphical user interfaces is to reify, i.e. make things visible, give representation to abstract notions. Collaborative interfaces could reify tasks, plans and goals, thus allowing both the system and the user direct access to shared goals and plans on how to achieve said goals. In order to mimic human-human collaboration, computer systems must be capable of some reasoning about and representation of mental states, actions, etc. Interaction and direct feedback allow the user and system to incrementally provide

relevant information for collaboration, enriching the limited reasoning capabilities of systems. Natural communication should enable both humans and computers to communicate intuitively via natural language, visuals, gestures, touch, direct manipulation, etc. Finally, collaborative systems must evolve together with their users. Based on user behavior, the system should automatically adapt step-by-step, with user feedback taken into consideration at each step, making adaptation a collaborative endeavor.

Grosz [4] reiterates the importance of reification. To enable collaborative functioning, computer systems should "integrate their communications capabilities with their underlying functionality" and be equipped to implicitly or explicitly access user plans and goals. This allows users to interact with the system in a more sophisticated, goal- and task-related manner, rather than just instructionally [4]. According to Grosz [4], collaboration implies a common goal, commitment to achieving this goal, a "shared recipe" on how to achieve the goal, commitment to the success of other participants, and a way to communicate.

Das [2] sees human-computer collaboration as a partnership between the cognitive and the computational processes. Computer systems are very good at computation, which is based on algorithms and precise rules. Often, however, complex tasks cannot be solved through algorithms because no such algorithms may exist or simplified algorithms produce significantly different results from what was intended. Cognitive process, on the other hand, is not restricted by precise rules and can draw upon subjective experiences, feelings, prejudices, etc. to reach solutions not available to purely computational approaches. Thus, in solving complex problems, a strategic cooperation between computation and cognition is necessary to achieve optimal results effectively. Das [2] clearly supports the "human complementary" approach to collaborative interfaces, stressing the importance of dividing information processing tasks according to the asymmetric strengths of the partners.

Similarly, Shieber [13] argues that collaborative human-computer interaction can be achieved by leveraging user efforts through a more equal division of labor according to partner capabilities. Designing for collaborative interfaces, then, requires a representation of the overall task that is easily decomposable into portions suitable for different problem-solving skills.

Rich and colleagues [12] seem to agree that flexible and adaptable distribution of tasks is a characteristic of intelligent user interfaces. However, they propose that an intelligent user interface "mimics the relationships that typically hold when two humans collaborate on a task involving a shared artifact," thus clearly focusing on emulating human-like capabilities in computer systems. In accordance with the other authors, they agree that collaboration implies common goals, pooling of capabilities and resources, communication and coordination [12].

Designing for Collaboration

Based on well-known and widely accepted usability heuristics [11], the paper will now examine how human-computer collaboration incorporates usability. There is not enough space here to do justice to all ten usability heuristics. However, the three heuristics included in the following discussion should provide

some insight into the perspective of designing for collaboration and how it entails usability.

Nielsen's first heuristic is about visibility – it is essential to keep users informed about what is going on with the system at all times through appropriate and timely feedback [11]. Designing for collaboration stresses the importance of reification, making visible and manipulable system status, communication messages, user tasks, plans and goals as well as other collaboration-relevant knowledge. Reification is the basis for successful communication and the establishment of a shared goal in human-computer collaboration. The usability heuristic of system status visibility is thus a small part of the communication strategy in a collaborative interface. For example, imagine a writer's aid similar to the one proposed in [1]. According to Nielsen's heuristic, when the system is searching for a specific citation, it should inform the user about its activity through a status message. According to the collaboration paradigm, the system should not only display a status message, it should also make clear and manipulable what the system is searching for, how is it doing the search and how to improve the search when necessary. Furthermore, the system could also help the user keep track of his/her own progress within a task. For example, the system may show how many citations in a paper have been completed and how many still need user input as well as highlight the incomplete citations. The system should do all this without directly interfering with the user's work.

The heuristic called "flexibility and efficiency of use" [11] caters to the needs of more experienced users. Interfaces should allow users to tailor their frequent actions to best suit their habits and skills. In designing for collaboration, adaptation becomes a collaborative endeavor in itself. User input, such as customizing frequent actions, is mixed with automatic system adaptation based on personal usage patterns, with one feeding into the other. Continuing with the example of a collaborative writer's aid – the usability heuristic requires that the system, among other things, enable users to create shortcuts to their favorite actions, such as search by (…), etc. Designing for collaboration should allow, besides users creating shortcuts, also the system to propose defaults, such as search by keyword, if that is the most frequently used option. The final personalized version of the writer's aid is based on both user and system input, with the user being able to change system proposed defaults.

"Match between system and the real world" heuristic focuses on the intuitiveness of the system. It is essential for usable systems to speak the user's language, instead of using technical terms and jargon. Information should be presented in a natural and logical order, corresponding to the user's expectations. In collaborative systems, the focus is on natural bi-directional communication between the user and the system, thus, it is important that the partners share a mutually suitable language. Let us take the writer's aid as an example again. A usable writer's aid would use common and sensible functions for commands (search, show, cancel), and present specific options in a habitually expected manner (drop-down lists, visual aids). A collaborative writer's aid would do the same, but in both directions and using more sophisticated interaction forms than just button-clicking and field-filling. For example, suppose the system has found a correct citation, the user accepts it and decides to

search for more similar papers. Intuitively, the user could just say "Similar papers" and the system would either find similar papers by author, year, topic, etc. or let the user clarify the task. Alternatively, on-demand options (such as tool tips) associated with the author, year or topic could appear, that allow the user to pick the appropriate task. The communication style is suitable as long as it is effective and efficient for both partners.

This work in progress is aimed at being the first attempt in building a link between usability and human-computer collaboration. Much work remains to be done in refining the theoretical framework behind designing for collaboration and in establishing its usefulness for complex systems design. Furthermore, empirical evidence in support of the hypothesized link between usability and collaborativeness is currently lacking.

Acknowledgements

This material is partly based upon work supported by the National Science Foundation under Grant No. 0819333. Any opinions, findings, and conclusions expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Citations

[1] Babaian, T., Grosz, B. J., & Shieber, S. M. A writer's collaborative assistant. *Int. Conf. on Intelligent User Interfaces*, (2002), 7-14.

[2] Das, B. Cognition friendly interaction: A concept of partnership in human computer interaction. *Chaos (Woodbury, N.Y.)*, *11*, 3 (2001), 632-640.

[3] Gersh, J. R., McKneely, J. A., & Remington, R. W. Cognitive Engineering: Undertanding Human Interaction with Complex Systems. *Johns Hopkins APL Technical Digest*, *26*, 4 (2005), 377-382.

[4] Grosz, B. Beyond mice and menus. *Proc. of the American Phil. Soc.*, *149*, 4 (2005), 529–543.

[5] Haynes, S. R., & Kannampallil, T. G. Learning, Performance, and Analysis Support for Complex Software Applications. *Proc. of the 3rd Ann. Workshop on HCI Research in MIS*, (2004), 30-34.

[6] International Organization for Standardization. ISO 9241-11:1998 Ergonomic Requirements for office work with visual display terminals – Part 11: Guidance on Usability. (1998).

[7] Law, E., Vermeeren, A., Hassenzahl, M., & Blythe, M. Towards a UX manifesto. *Proc. of the 21st British HCI Group Ann. Conf. on HCI 2008: People and Computers XXI: HCI... but not as we know, 2, (2007)*.

[8] Licklider, J. C. R. Man-computer symbiosis. *IRE Transactions on Human Factors in Electronics*, 1, (1960), 4-11.

[9] Mirel, B. *Interaction Design for Complex Problem Solving*. San Francisco, CA: Morgan Kaufman (2004).

[10] Nielsen, J. *Usability engineering*. Boston, MA: AP Professional (1993).

[11] Nielsen, J. Ten Usability Heuristics. Retrieved from http://www.useit.com/papers/heuristic/heuristic_list.html

[12] Rich, C., Sidner, C., & Lesh, N. Collagen: applying collaborative discourse theory to human-computer interaction. *AI magazine*, *22*, 4 (2001), 15–26.

[13] Shieber, S. M. A call for collaborative interfaces. *ACM Computing Surveys (CSUR)*, *28*, 4 (1996).

[14] Soloway, E., Guzdial, M., & Hay, K. E. Learner-centered design: the challenge for HCI in the 21st century. *Interactions*, *1*, 2 (1994), 36-48.

[15] Terveen, L. G. Overview of human-computer collaboration. *Knowledge-Based Systems*, *8*, 2-3 (1995), 67-81.