Improving ERP Usability Through User-System Collaboration

Tamara Babaian
Computer Information Systems Department
Bentley College
Waltham, MA 02452-4705
(781) 891-3161
Fax: (781) 891-2949
tbabaian@bentley.edu

Wendy Lucas
Computer Information Systems Department
Bentley College
Waltham, MA 02452-4705
(781) 891-2554
Fax: (781) 891-2949
wlucas@bentley.edu

Heikki Topi
Computer Information Systems Department
Bentley College
Waltham, MA 02452-4705
(781) 891-2799
Fax: (781) 891-2949
htopi@bentley.edu

Abstract

Anecdotal evidence strongly suggests that enterprise resource planning (ERP) systems have unintuitive user interfaces that hinder usability, frustrate users, and ultimately interfere with their successful adoption and utilization in organizations. Despite the huge costs associated with poorly implemented systems, ERP usability has received little attention from the IS and HCI research communities. We argue in this paper that existing theories on usability should be extended to address the unique challenges resulting from the size, complexity, and integrated functionality of these industrial behemoths. We believe that collaboration theory is a new and beneficial way to conceptualize the relationship between the user and the system. This theory has the potential to provide a foundation for user-system interaction that enhances user performance and satisfaction with ERP systems.

Keywords: Usability, Enterprise systems, ERP, Collaboration theory, Collaborative user interfaces

Introduction

A recent study by Forrester Research (Chew, Orlov, & Herbert, 2003) evaluated eleven different ERP products, including SAP, PeopleSoft, Oracle, JD Edwards, Microsoft and Lawson. Its findings suggested that poor usability characteristics and the unintuitive user interfaces of these systems contribute to decreased productivity and increased costs for businesses using them. In trying to perform a number of standard tasks that should have been "straightforward" without any training, the analysts from Forrester found that several of these tasks required "inordinate patience and expertise" to complete (Gilbert, 2003). The overall conclusion was that "users should demand better usability." Yet, there has been little movement to date toward improving the design of the user interface components of these systems by either ERP vendors or the usability community as a whole. This motivates the research initiative described here.

Given the time, effort, and money expended on implementation and training, it is surprising that so little attention has been focused on understanding the ways in which users interact with ERP software and the degree to which the interaction model supports the tasks being performed. In this paper, we suggest that applying the principles of collaboration (Bratman, 1992) to systems development provides a means for addressing the gap between the capabilities of the ERP system and harnessing those capabilities to meet each user's individual objectives. By "collaboration," we refer to the collaboration between the user and the system, as opposed to using computing technology to support collaboration between users, which is commonly referred to as computer-supported cooperative work (CSCW). The novelty of our research lies in its emphasis on the relationship between collaborative support, task performance, and satisfaction. We believe that the more aligned the technology is with the user's goals, the

better able it will be to respond in a collaborative manner to the user's needs, enhancing both task performance and satisfaction with the process.

Our long-term research goal is to improve the usability of enterprise systems by increasing the collaborative capabilities of their interfaces. This research currently includes the following components:

- Field studies focusing on the nature of the users' everyday needs and interactions with these systems
- Development of enterprise system design guidelines based on collaboration theory
- Development of interface evaluation techniques based on collaboration theory
- Implementation of prototype ERP interfaces for validating the design and evaluation methodologies we are developing

In this paper, we elucidate the role of collaboration theory in our research and illustrate the benefits gained by using it for ERP design and evaluation.

The rest of this paper is organized as follows. The next section describes the principles of collaborative behavior and illustrates how they can be used for establishing guidelines for usability design and evaluation. Then we present a review of related literature. Next, we provide an example scenario of a user performing a typical ERP task and demonstrate how the interface being used could be improved by taking a collaboration-based approach to its design. The last section draws conclusions and outlines future work to be done in this area.

Collaboration Theory for Interface Design and Evaluation

The core thesis of this paper is that collaboration theory can be applied as a set of guiding principles to the design and evaluation of ERP systems. In this section, we discuss the overall

characteristics of this theory and illustrate how taking a collaborative view of user-system interactions influences the design and evaluation processes and leads to enhanced system usability.

Grosz (1996) and Shieber (1996) suggest that human-computer interaction should move from a master-slave model, in which the human user issues commands to the system, to a model based on collaboration between the system and the user in order to provide an adequate level of support to users in the increasingly complex environments of modern applications. In other words, the computer system should be designed to act as the user's partner in the process of goal achievement. This view of a system-collaborator, supported by a philosophical account of cooperative activity (Bratman, 1992) and by more formal mathematical frameworks for such a collaboration (Grosz and Kraus, 1996), has already been used in the design and implementation of several prototype interfaces in the intelligent agents community (e.g., Babaian, Grosz, and Shieber, 2002; Rich, Sidner, and Lesh, 2001). None of these, however, have been interfaces to enterprise-wide administrative systems.

It should be noted that the phrase "system-partner" is not to be taken literally here. Computing technology does not yet have the capability to implement a collaboration partner with human-like abilities, which would allow it to observe the process, recognize the user's intentions on the fly, "jump-in," and automatically solve the problem. Rather, moving towards more collaborative behavior on the part of the system can be accomplished by a careful design based on the principles of collaboration. As defined by Bratman (1992) and further elaborated for computational use by Grosz and Kraus (1996), these principles are:

Commitment to the joint activity. Each party recognizes the joint activity and is committed to it. As part of this commitment, the parties need to be aware of the context

surrounding their collaboration because it may be important in determining the finer details of that activity.

Mutual responsiveness. Each participant seeks to adjust his behavior based on the behavior of the other and guided by his commitment to the joint activity. Mutual responsiveness, in conjunction with this commitment, means that the parties may have to adapt their actions for the benefit of the more optimal joint outcome.

Commitment to mutual support. Each party is committed to supporting the efforts of the other. When an agent knows the other party may need help in performing a subtask related to their shared activity and is able to provide such help, the agent is ready to assist and the other party recognizes and supports such assistance. Commitment to mutual support also implies communication with the purpose of sharing information that is essential for the completion of the joint activity.

Meshing subplans. The parties should seek to decompose the task into mutually meshing, although independent, subplans. The parties must thus engage in communication to coordinate their independent subplans at certain times, as the need arises.

To illustrate how these principles can change the approach taken to the design and evaluation of systems, we first describe a well-known usability evaluation method called the cognitive walkthrough (Wharton et al., 1994), and then show how taking a collaborative view of user-system interaction would affect it. The basis of the cognitive walkthrough method is a theory of exploratory learning called CE+ (Polson and Lewis, 1990), which was developed to guide the design of interfaces that are easily learnable. This work therefore bears similarity to our proposed usage of collaboration theory as a set of guiding design and evaluation principles.

A cognitive walkthrough involves an analyst evaluating an interface by creating a scenario of its usage for a particular task. During the walkthrough, the following questions, taken directly from Wharton et al. (1994, p. 112) must be answered in order to assess an untrained user's success in invoking the appropriate system action at each step of the way. The answers to these questions then form a basis for predicting a user's success or failure to properly complete the task.

1. Will the users try to achieve the right effect?
2. Will the user notice that the correct action is available?
3. Will the user associate the correct action with the effect trying to be achieved?
4. If the correct action is performed, will the user see that progress is being made toward solution of the task?

Taking a collaborative view of user-system interactions would affect the formulation of each of these questions. Keeping in mind that this view requires us to think of the system as an equal partner in the process, let us consider the first question ("Will the users try to achieve the right effect?"). It is designed to capture the user's ability to recognize the relationship between the structure of the task and the system's known functions. The example that Wharton et al. (1994) use to illustrate this question is a user whose task is to print a document, but who cannot achieve this goal without first selecting a printer. Will the user know that his next immediate step should be to select the printer?

Evaluating this situation from the collaboration perspective brings us to a different conclusion regarding the relevant question to ask. It is not whether or not the user knows that the next step is to select the printer. Rather, it is whether or not the system has been designed in such a way that it knows the next step in the printing process is the selection of a printer and will act

in accordance with this knowledge. The system must therefore be aware of the overall "recipe" for printing, which connects the two actions of selecting the printer and sending the document to it. Once the user has identified the goal of printing a document, the system should proceed with an action that enables the user to select a printer, either by choosing one from a system-generated list or by specifying a new one. It is the commitment to mutual support that causes the system to aid the user with the printer selection process. It would not be collaborative for the system to send the document to any printer without first consulting the user for a number of possible reasons, such as the user must know where to pick up the printed document and the default option may not the best choice in this particular instance.

Thus, the first question can be modified according to the principles of collaboration to read as follows:

1. Based on the user's overall goal, will the system recognize the next step in the process and either act to perform that step or, if the user's input is necessary, present a set of alternative actions from which the user may make a selection?

A similar viewing of the three remaining questions in light of collaboration theory leads us to the following possible versions of questions two through four:

2. Does the system help the user identify the next action and present it in a highly visible manner?
3. Does the system present a meaningful set of alternative actions based on the user's overall goal?
4. Will the system keep the user informed about the consequences of actions taken by either the user or the system, as they relate to progress made toward the achievement of the task?

In transforming the walkthrough questions to reflect the collaborative view of system-user interaction, we have shifted the focus from the actions, knowledge, and capabilities of the user alone to include the system as an equal party in the process. If a system is designed with this view in mind and evaluated using questions based on the principles of collaboration, the nature of the relationship between the user and the system will change. We believe that the collaborative view of the user-system relationship will result in interactions that allow users to achieve their goals with less effort and frustration and more accuracy, due to the additional support provided by the system.

While the example we have used in this evaluation (namely, selecting a printer prior to printing) is a very narrow task compared to many of those that are encountered in the enterprise systems environment, it makes the point that even the simplest of tasks can benefit from the application of the principles of collaboration. The benefits of user-system collaboration would be significantly greater with more complex organizational tasks, as the system could provide knowledge and assistance for those cases where the correct sequence of events is not readily discernable by even the most educated of users. Given the lack of transparency in performing enterprise system transactions, an approach that sheds light on the recipes for successful task completion holds great promise for improving user productivity through enhanced system usability.

In the following section, we discuss the most important approaches that have been followed to date for usability design and evaluation in the fields of human-computer interaction and enterprise systems. We then position the collaborative view in the context of these approaches.

Related Research

A review of current literature yields few studies that discuss interface design or usability in the context of enterprise systems. Bishu et al. (1999) do, however, raise some of the human factors implications of ERP systems, including the lack of attention paid to training and the maze of screens one has to navigate. A recent annotated bibliography on ERP system research by Esteves and Pastor (2001) does not include any references to papers that directly discuss ERP usability. Similarly, a comprehensive collection of state-of-the-art articles on usability (Jacko and Sears, 2003) does not include any articles discussing usability issues of ERP systems, enterprise systems, or any other type of administrative organizational system. While there is little research addressing ERP usability, research on human-computer interaction in general has made considerable advances, as evidenced by collections such as Jacko and Sears (2003) (see above), and a large number of innovative interface types. Although a number of experimental interfaces have even found their way into practice, they have not been used in the context of ERP systems, to the best of our knowledge. Applying the scientific and technological advancements that have been made in user interface research to these systems holds great promise for improving their usability.

It is virtually impossible to create a highly usable system without addressing usability issues from the start, that is, at the requirements analysis and design stages (Maguire, 2001). This approach is known as user-centered design. Although methods employed for this type of design differ significantly in their underlying theories, the key component of all modern usability design and evaluation techniques is a clear understanding of the user's goals and tasks. These goals and their associated tasks can be of different granularities, ranging from broadly defined ones like "retrieve, relate, and report financial, production, and personnel data in order to persuade [a]

manager to allocate effort and resources differently" (Mirel, 1996, p. 16), to very specific ones involving a few clearly articulated changes to an existing document.

The broad scope of ERP and other enterprise-wide systems creates its own special requirements for usability analysis. Rather than evaluating usability one function at a time, it is necessary to analyze the integrated use of the multiple system features required for achieving a comprehensive goal. Few existing usability methodologies are appropriate for this type of analysis. Model-based methodologies such as GOMS (Card et al., 1983), for example, cannot be applied due to the obvious difficulty of specifying complex tasks and respective user behaviors at a detailed level. While task analysis (Redish & Wixon, 2002) can be used to model a hierarchy of goals and tasks at a high level, it does not address the interactions between the system and its user. Therefore, task analysis cannot be effectively used in the design of user interfaces. Although it is theoretically possible to evaluate the interactions involved in performing a comprehensive task using inspection methods (e.g. Nielsen, 1993; Wharton et al., 1994), Cockton et al. (2002, p. 1121) report that this kind of verification is overwhelmingly left out of usability evaluations. This can be attributed to the fact that the guidelines on which the methods are based do not address the dynamics of the interaction between the system and its user, but rather focus on the more static aspects of the system.

User-based methods (Dumas, 2003) are great at uncovering usability problems, but they are focusing on specific features of the existing implementation. They therefore tend to elicit information that is boxed within the framework of the specific tool being evaluated, leading to localized fixes rather than system-wide alterations of the design.

The collaboration view of a system-user interaction offers a powerful alternative to the above approaches by explicitly including the user and the system in the single model of

interactions involved in completing the task. This changes the dynamic from the user being the only one with responsibilities and knowledge about the process to one that incorporates the system as well. The system's role is expanded to include helping the user perform tasks, as it is no longer limited to just responding to commands. Collaboration is supported based on the system's and user's knowledge of the process flow. While existing techniques of task analysis and modeling focus on the hierarchical structure of individual tasks, collaboration theory offers a comprehensive view of the entire system. This naturally leads to specific requirements regarding the knowledge and behavior of the system, which is acting as a partner in performing these tasks.

Transaction Task Example

To illustrate the use of collaboration principles in the context of an ERP system, we apply these principles to a common ERP transaction task. In this walkthrough scenario, we point out how a fictitious materials management system that closely resembles a well-known and widely-used ERP system fails to support the user in achieving her goal. In the subsequent discussion, we suggest how that system could be modified to be a better collaborator. This is followed by a snapshot of our prototype implementation and a description of how collaboration theory being used to influence its design.

Scenario

> *Pat is an engineer and a relatively new user of a large enterprise system. As part of her engineering assignment, Pat needs to order a certain hardware component. She tries to create a purchase requisition, but is stymied when she can't specify the item to be ordered because it is not listed in the Material Master.*

The option of adding a new part to the Material Master is not available in the purchase order interface, although its implementation exists and is available elsewhere in the system. Interface design based on collaboration should recognize the broader context in which the task of creating a purchase requisition may occur. Based on the mutual support principle, the system should provide easy access to related or prerequisite tasks, such as adding a new part in the context of a purchase requisition.

> *Pat has to scrap the unfinished purchase requisition, enter the part into the Material Master, and then proceed to create the purchase requisition again. To create a new purchase requisition, Pat follows this menu path: Logistics – Material Master – Purchasing – Purchase Requisition. – Create. She enters information regarding the delivery date, the plant to which this part must be delivered, the storage location, and the purchasing group.*
>
> *When Pat presses Ok to move to the next screen, the system complains: "Date period D is not valid." Pat goes back to the date field and tries to modify the date specification. Reading the system help on various formats fails to explain how the D, T, W, or M options affect the format of the date to be entered (particularly since Pat does not recognize that the use of the letter 'T' for 'Date' might not be based on the English language). She remains puzzled for a while until she stumbles upon the Possible Entries option that is available for the date field. Selecting this option results in the system displaying a calendar from which Pat selects a date, which is then correctly entered for her by the system into the date field.*

Although the interface includes the very useful option of selecting the date from a calendar, this option is not offered and remains obscured even though the system has detected

and reported the user's error. Commitment to mutual support and mutual responsiveness would require a system-collaborator that has the ability to offer such help when it can provide it, instead of merely informing the user about a failure.

A colleague then suggests that Pat select the Model service specifications option, which displays the actual names of all items listed in the form in addition to their numeric identifier. Pat finds this option very helpful for both clarity and verification purposes, and opts to use it.

Commitment to mutual support requires that the collaborating parties share the knowledge that is relevant to the success of the joint activity. In the previous example, even though displaying the item names in addition to the identifiers would be more informative from the perspective of a human user and is very easy for the system to do, the interface does not provide this information without a specific request. Typically, new users are not aware of all of the available options, and thus fail to take advantage of these types of capabilities.

Pat verifies that the information she has entered, including the destination plant for the part, is correct, confirms this to the system, and is taken to the next screen, where she is asked to list the items to be ordered. Unfortunately, Pat has forgotten the exact ID number of the part she just entered into the Material Master.

If the system kept track of the steps Pat had taken previously, it could use this information to examine the context of the current interaction. It would then be able to recognize that, having just entered a new part, Pat is likely to need to refer to this part's information when she follows up with the purchase requisition.

Pat tries to find the ID number by reviewing the item descriptions using the Possible Values option for the item field. At some point during this review process, the

information on the screen changes completely. Pat is unsure what she has done to cause this change and wonders whether or not the information on the purchase requisition is still available.

The rapid and drastic change on the screen creates an impression that the purchase requisition task has been abandoned. Pat is now unsure of whether the system is still committed to the joint activity of creating a purchase requisition. This situation demonstrates the need for the system to convey the future steps (plan) in performing the task as well as the history of the steps performed and the context of the most current interaction. Collaborators need to communicate in order to make sure their mutual plans for achieving the shared goal are coordinated.

After an initial moment of panic, Pat discovers that she can still get to the list of items in the purchase requisition by using the Go Back button, and heaves a sigh of relief.

There would be no panic if Pat knew exactly where she was in the process. She should be kept aware of the plan by the system-collaborator and be able to get back to the previous steps.

There are more than 12 available options for displaying the material lists – too many for Pat to make use of them all.

Pat has just provided the system with information regarding the plant for which the part is being ordered. The system should be able to infer that the list of parts for this plant should be most useful for the search, and perhaps rate that option higher than other searches for parts.

Feeling overwhelmed by choices, Pat finally notices an option for displaying parts by plant and, in reviewing the material list for the destination plant, locates the

description of the part there. Upon specifying the quantity, Pat is done creating this document. She feels unsure, however, if the information she has entered is complete because there are a lot of other fields on the form that she has left blank. After consulting the help desk, Pat concludes that the purchase requisition is complete and saves it.

The system knows which fields are optional. It should communicate this knowledge clearly to the user because this would help Pat complete the task with confidence.

Discussion

As is evident from the above scenario, the numerous data and process dependencies built into an ERP system are largely hidden from the user. Coupled with the vastness of these systems, it is virtually impossible for any user to know about all, or even most, of these dependencies. Yet, users are often required to be familiar with at least some of these dependencies in order to fully understand the ramifications of their actions, determine what the next step should be in order to carry out a business process, or diagnose and fix an error. The mutual responsiveness principle of collaboration would require that the system shares with its users any information that is necessary for the achievement of their goals in a clear and effective manner. The system is better equipped than a human user for "remembering" such a large and complex set of relationships spanning multiple domains. Thus, the system that is committed to supporting its users should assume responsibility for guiding them through the interrelated processes and helping them find the relevant data. One way to achieve this is by having the system make the recipes for complex business tasks available to its users. Moreover, the system must be designed to be aware of the broader business context of each user-computer interaction in order to

recognize (figuratively speaking) the high-level goal that the user is trying to achieve and guide her through the multiple-step business process.

In the case of human errors and uncertainty over how to proceed (such as Pat's confusion over how to correctly enter the date), it is critical that an ERP system clearly communicate possible causes and/or alternative courses of action instead of simply reporting that an error has occurred. Effective communication is a hallmark of successful collaboration, and mechanisms must be built in for enabling clear communication from the system to the user. To be truly effective, this communication should be based on the business vocabulary employed by the organization.

The scenario also shows that the principles of collaboration influence the design of both the static components of the interface and the dynamic elements resulting from the human-computer interaction. Including an "*Add new part*" option to the purchase requisition interface and displaying information about which fields are required are examples of modifications to the static components of the interface. Recognizing that the number for a newly added part may be used in the purchase requisition that follows is an example of keeping track of a dynamic element and considering the broader context for each simple interaction. Collaboration principles should be used to guide system behavior in these static and dynamic contexts.

Illustrative Prototype

We are currently working on the design and implementation of a prototype involving several categories of ERP tasks for demonstrating how collaboration principles can be used to create efficient and usable interfaces to enterprise tasks. The snapshot shown in Figure 1 illustrates some of the important features of that interface.

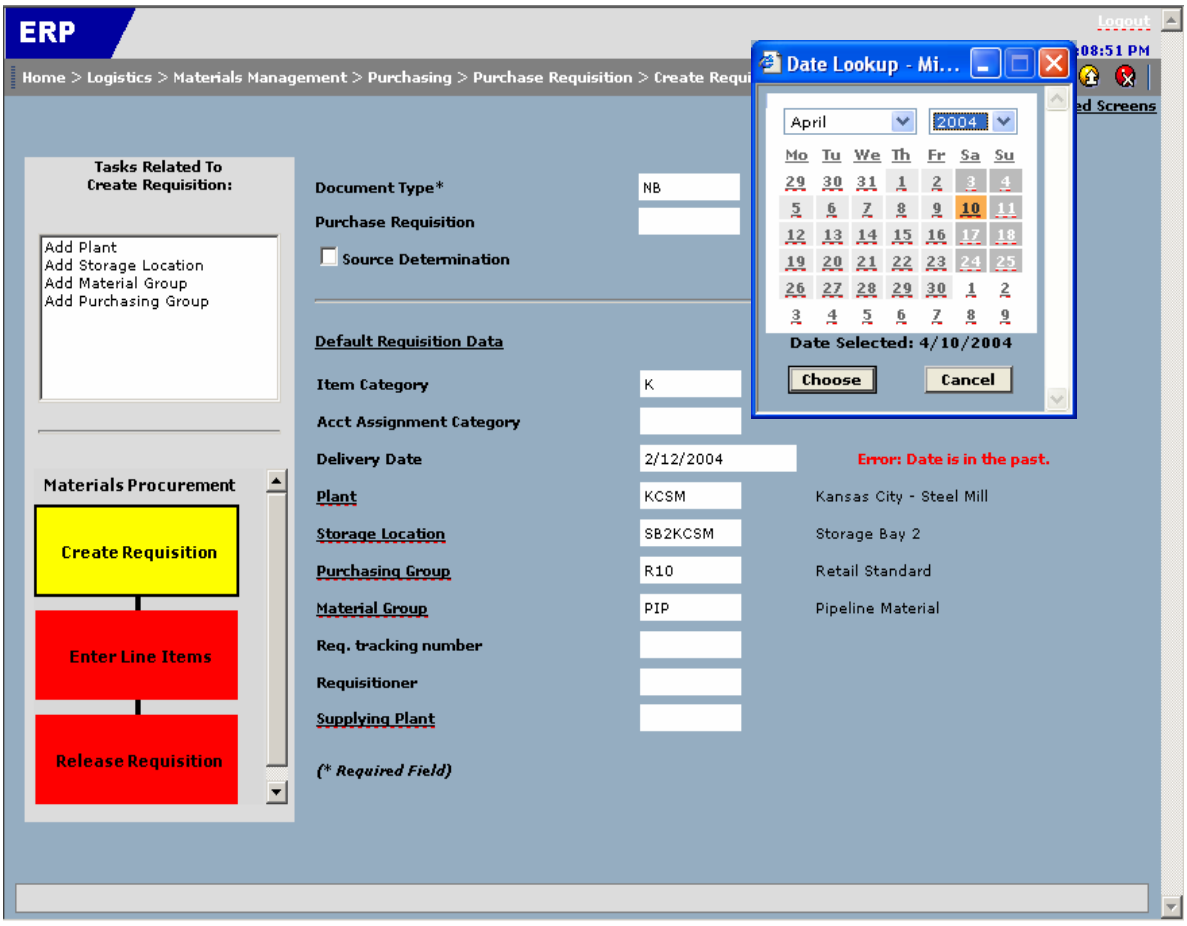

Figure 1: Prototype Implementation

This figure captures a moment in the process of the user creating a purchase requisition (PR). The main screen displays a set of fields for which the user can specify values. Required fields are noted with an asterisk. Some field values have been entered by the system automatically, based on the information it has about the user and the plant (note that the user is free to change these values if they do not correspond to the actual parameters for this PR). Field names that appear to be underlined are actually links to related tasks (for example, the "*Plant*" field links to the "*Add Plant*" option). The system also displays a set of links to related tasks on the left hand side of the screen, which provide easy access to the data and processes that are closely related to creating a PR.

The text in red next to the Delivery Date field shows that the system has noticed an error; namely, the value entered by the user refers to a date in the past. In addition to notifying the user of this error, the system has simultaneously displayed a calendar so that the user can easily correct the mistake by selecting another date.

The steps required by the process for creating a PR are depicted in a flowchart on the left, with the current step appearing in yellow with a black border. In addition to informing the user about the process, this chart also provides easy access to previous and subsequent steps. While a user can skip to a later task, he can only view that task; all preceding steps must be completed before any data can be entered.

This prototype demonstrates some simple steps that were taken to enhance the usability of an ERP interface for one particular task based on the principles of collaboration. It is important to note that the greatest value of applying these principles to ERP interfaces will come from the system's ability to support users in performing a broad range of tasks spanning multiple business processes.

Conclusions and Future Work

Improving the usability of ERP systems provides benefits that extend well beyond meeting the needs of individual users. It benefits the organization as a whole by reducing the length of the training time, improving employee satisfaction, and providing valuable information on overall system usage. This paper has argued that collaboration theory is a highly relevant conceptual framework that can be used effectively to guide the design and evaluation of user-system interaction in the context of large-scale enterprise systems. While the examples we have shown here for improving user interaction do not go beyond those existing separately in

individual implementations, it is the application of the principles of collaboration for methodically addressing system usability that is the unique contribution of this paper.

It has been argued by usability researchers as well as researchers in the collaborative interfaces community that design for usability cannot be achieved by a local change in the interface. Collaboration cannot be "patched on" and must be designed in from the start. The influence on the design is not limited to the system's front-end: to implement the collaborative nature of the interaction generally requires appropriate support in the data model and the algorithmic modules of a system.

Investigating the design principles and the resulting representational and algorithmic needs stemming from the user-system collaboration model of the interface is especially interesting and important in the context of enterprise-wide systems, and not only because of the obvious shortcomings of ERP interfaces. These systems span an enormously broad domain of organizational tasks, with most tasks involving multiple logical and physical system modules. In addition, there are multiple users with varying demands and levels of expertise. All of these factors increase the challenge of enhancing system usability.

We believe that collaboration theory is an excellent foundation for usability design and evaluation because:

- It directly addresses the process of collaborative problem solving in a systematic way by suggesting a set of requirements and procedures that must be in place to achieve successful collaboration.
- It provides a framework for analyzing many existing user interface practices and developments that improve system usability and helps in explaining their benefits.

- In addition to its role as an evaluation framework, it can simultaneously be used to guide design choices. This has been identified as one of the challenges of usability research by John (1996).

As of this time, we have applied the principles of collaboration to the design and evaluation of enterprise systems through the use of scenarios and case examples. This approach appears to offer great promise outside of the relatively narrow set of domains where it has already been applied, as noted earlier. Building on the approach suggested by Wharton et al. (1994), we are currently developing a formal collaborative walkthrough method for usability evaluation. Further testing of this method will undoubtedly result in its fine-tuning. We plan on evaluating the predictive power of our collaborative walkthrough method by conducting experiments using our prototype implementation.

One of the core ideas of this paper is that large-scale enterprise systems are a particularly useful domain for applying the principles of collaboration to user-system interactions. To explore the validity of this claim, it is important to conduct systematic field studies that focus on the users' perceptions regarding the usability of enterprise systems. In-depth case studies based on interviews and observations as well as surveys should be used to improve our understanding of the factors that affect usability perceptions. Field-based research, together with laboratory studies, will allow a comprehensive evaluation of the opportunities that collaboration theory offers for improving the usability of enterprise systems.

Future research on ERP and enterprise system usability should address both the technical issues related to user interface design as well as the overall impact of ERP interfaces on organizational decision-making.

Acknowledgments

This work was funded by a grant from Bentley College. We gratefully acknowledge this support. Brian Bento worked on the development of the prototype. An earlier version of this paper appears in the Proceedings of ICEIS'04. We appreciate the feedback given by the conference referees and participants.

References

Babaian, T., Grosz, B. J., & Shieber, S. M. (2002). A writer's collaborative assistant. Paper presented at the 2002 *International Conference on Intelligent User Interfaces* (IUI-02), New York, NY.

Bishu, R. R., Kleiner, B. M., & Drury, C. M. (2000). Ergonomic concerns in enterprise resource planning (ERP) systems and its implementations. Paper presented at the *Fourth International Conference on the Design of Information Infrastructure Systems for Manufacturing* (DIISM 2000), Melbourne, Victoria, Australia.

Bratman, M. E. (1992). *Shared cooperative activity*. The Philosophical Review, 101(2), 327-341.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale: Lawrence Erlbaum.

Chew, J., Orlow, L., & Herbert, L. (2003). App user interfaces still need work, a technology brief by Forrester Research. Retrieved Jan 26, 2004, from http://www.forrester.com/ER/Research/Brief/Excerpt/0,1317,16184,00.html

Cockton, G., Lavery, D., & Woolrych, A. (2003). Inspection-based evaluations. In J. A. Jacko & A. Sears (Eds.), *The human-computer interaction handbook: Fundamentals, evolving technologies, and emerging applications*. Mahwah, New Jersey: Lawrence Erlbaum.

Dumas, J. S. (2003). User-based evaluation. In J. A. Jacko & A. Sears (Eds.), *The human-computer interaction handbook: Fundamentals, evolving technologies, and emerging applications*. Mahwah, New Jersey: Lawrence Erlbaum.

Evestes, J., & Pastor, J. (2001). *Enterprise resource planning systems research: An annotated bibliography*. Communications of the AIS, 7(8).

Gilbert, A. (2003). Business apps get bad marks in usability. Retrieved Oct 4, 2004, 2004, from http://news.com.com/2100-1017-980648.html

Grosz, B. J. (1996). Collaborative systems. *AI Magazine*, 17(2), 67-85.

Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence,* 86(2), 269-357.

Jacko, J. A., & Sears, A. (Eds.). (2003). *The human-computer interaction handbook: Fundamentals, evolving technologies, and emerging applications*. Mahwah, New Jersey: Lawrence Erlbaum.

John, B. E. (1996). Evaluating usability evaluation techniques. *ACM Computing Surveys*, 28(4), 139-139.

Maguire, M. (2001). Methods to support human-centred design. *International Journal of Human-Computer Studies*, 55(4), 587-634.

Mirel, B. (1996). Contextual inquiry and the representation of tasks. *The Journal of Computer Documentation*, 10(1), 14-21.

Nielsen, J. (1993). *Usability engineering*. San Diego, CA: Academic Press, Inc.

Orbitz, C. L., & Grosz, B. J. (2002). Interpreting information requests in context: A collaborative web interface for distance learning. *Autonomous Agents and Multi-Agent Systems*, 5(4), 429-465.

Polson, P., & Lewis, C. (1990). Theory-based design for easily learned interfaces. *Human-Computer Interaction*, 5, 1-48.

Redish, J., & Wixon, D. (2003). Task analysis. In J. A. Jacko & A. Sears (Eds.), *The human-computer interaction handbook: Fundamentals, evolving technologies, and emerging applications*. Mahwah, New Jersey: Lawrence Erlbaum.

Rich, C., Sidner, C., & Lesh, N. (2001). COLLAGEN: Applying collaborative discourse to human-computer interaction. *AI Magazine*, 22(4), 15-26.

Shieber, S. M. (1996). A call for collaborative interfaces. *ACM Computing Surveys*, 28(4), 143-143.

Wharton, C., Rieman, J., Lewis, C., & Polson, P. (1994). The cognitive walkthrough method: A practitioner's guide. In J. Nielsen & R. L. Mack (Eds.), *Usability inspection methods* (pp. 105-141). New York, NY: John Wiley & Sons.