A Collaborative Model of Feedback in Human-Computer Interaction

Manuel A. Pérez-Quiñones
Human-Computer Interaction Laboratory
Naval Research Laboratory, Code 5512
Washington, DC 20375-5337
perez@aic.nrl.navy.mil
+1-202-767-9003

John L. Sibert
Department of EE and CS
The George Washington University
Washington, DC 20052
sibert@seas.gwu.edu
+1-202-994-4953

ABSTRACT

Feedback plays an important role in human-computer interaction. It provides the user with evidence of closure, thus satisfying the communication expectations that users have when engaging in a dialogue. In this paper we present a model identifying five feedback states that must be communicated to the user to fulfill the communication expectations of a dialogue. The model is based on a linguistics theory of conversation, but is applied to a graphical user interface. An experiment is described in which we test users' expectations and their behavior when those expectations are not met. The model subsumes some of the temporal requirements for feedback previously reported in the human-computer interaction literature.

Keywords

Human-computer dialogues, feedback, conversational dialogues, states of understanding, collaborative view of conversations.

INTRODUCTION

Adequate feedback is a necessary component of both human-human and human-computer interaction. In human conversations, we use language, gestures, and body language to inform our conversational partners that we have heard and understood their communication. These communicative events fulfill a very important role in a conversation; they satisfy certain communication expectations of the dialogue participant. To understand these expectations imagine what happens when they are missing. In a conversation, if you do not hear a short uh hum or see a nod of the head, you might think that the other person did not understand and as a result you might interrupt the conversation while you question the listener's attentiveness; "hey, are you awake?"

This idea of communication expectations, also called "psychological closure" [16, 25], is a common human behavioral characteristic that also exists when we are interacting with a computer. For example, if the user types a command but receives no response from the user interface, the user might repeatedly press the return key to make sure that the system has "heard" (or received) the command. At times it is not even clear "who is waiting for whom" [17].

Many studies of feedback have been reported in the literature. Some deal with response times guidelines for feedback, others have addressed performance penalties caused by delayed responses [26], while others deal with the form feedback should take. What is lacking in all of these studies is a behavioral model that explains users' communication expectations and their behavior. In this paper we present such a model of feedback based on principles of conversation derived from linguistic theory and on graphical user interface design guidelines. We also describe an experiment in which we test some of the expectations proposed by the model.

BACKGROUND

Feedback, in most of the human-computer interaction literature, refers to communication from the system to the user as a direct result of a user's action [24]. Most user interface design guidelines also stress the importance of this type of feedback, for example [1] recommends "keep the user informed" and use feedback to confirm "that the operation is being carried out, and (eventually) that it's finished." Gaines [10] states that the response to a user's action "should be sufficient to identify the type of activity taking place."

However, feedback also can be used to communicate the state of the system independently of the user's action [1]. The system, at times, can be busy processing incoming events like mail messages, or alarm notifications that the user has preset days in advance. In these cases, the system must let the user know its current state of processing so that the user does not feel frustrated or locked out of the dialogue.

Foley and van Dam [8] describe feedback in terms of interpersonal conversations, such as responses to questions or "signs of attentiveness." Both of these conversational devices are classified as forms of feedback.

What is common in all of these views of feedback is that it serves a behavioral purpose in interaction. This purpose represents the communication expectations that humans have from a conversational participant, even when this conversational participant is a computer system.

Collaborative View of Conversations

Our model is based on Clark, et. al [5-7] who view a conversation between two humans as a collaborative process. Previously, conversations were seen as turns taken by two participants, where the turns were related by a series of adjacency pair relations. The collaborative view extends this turn-taking model by looking at the cooperation between conversational partners to agree on common ground, sometimes without taking extra turns.

The collaborative model states that conversations are really negotiations over the contents of the conversation between two participants. One participant, the speaker, makes a presentation and the other participant, the listener, accepts it. A *contribution* to a conversation occurs only after the presentation is accepted. It is defined to be the pair of presentation and acceptance turns.

This theory of contributions to conversations has already been applied in the study of human-computer dialogues. Payne [19, 20] analyzed the interaction in MacDraw based on presentation/acceptance trees. Brennan and Hulteen have defined a model of adaptive speech feedback [3] based on the states of understanding principles. A similar model is used in Apple Computer's speech input system [15], PlainTalk. The different states of feedback are identified with icons, as shown below.

sleeping listening hearing working

Figure 1. SOU in Apple's Casper

States of Understanding

States of understanding (SOU) are the states that a listener believes he or she is in, after receiving a presentation from the speaker [6]. SOU is one of many ways participants in conversations fulfill their goals in the collaborative process. The listener provides evidence of his or her SOU thus allowing the speaker to adapt according to the specific SOU.

The states are defined as follows. When speaker *A* issues an utterance, listener *B* believes he or she is in one of the following states:

State 0: *B* is not aware of the utterance
State 1: *B* is aware of the utterance, but did not hear it
State 2: *B* heard the utterance, but did not understand it
State 3: *B* fully understood the utterance

Evidence of understanding (i.e. identification of the state) must be provided by the listener, thus allowing the other participant to adapt accordingly. For example, if after presenting a question, the speaker receives evidence that the listener is in state 1, the speaker might repeat or rephrase the question to help the listener reach state 3. The two partners follow the principle of least collaborative effort [7]. This principle states that speaker and hearer will try to minimize the collaborative effort it takes to reach a level of understanding. If a SOU is not clearly communicated by the listener, the speaker will seek further evidence that the communication was received. This is a form of repair behavior guided by the SOU principles.

The SOU can be communicated without using a verbal utterance. For example, the use of a confused facial expression can be enough to indicate that the listener is in state 2. Also, several signals together can be used to signal a single state. A facial expression together with a hand gesture, for example, can be interpreted as a single indication of a state.

PROPOSED MODEL

Human behavior in a human-computer dialogue, when a communication expectation is not met, is similar to the repair behavior found in human-human dialogues. For example, consider the use of the performance meter on a workstation. Many workstation users keep one at the bottom of their screen to get an indication of the amount of work the CPU is currently performing. This information is used to identify whether the system is locked or just slow because it is overloaded. When users initiate an action but obtain no response within some expected amount of time, they face one of two situations: did the computer received my action or should I repeat it? The performance meter allows the user to determine if the system is busy, and thus may have received the action but is just slow in the response. If the system is not busy, the action was lost or ignored in which case the user must repeat the action if the results are desired. In this example, the user interface of the application in use did not meet the user's communication expectations and thus the user had to resort to an extra "device" (the performance meter) to find out extra information to match the communication expectations.

Based on the states of understanding principles and on graphical user interface design guidelines, such as [1], we have identified a number of feedback states needed in a human-computer dialogue to meet the human's communication expectations.

Our model contains five *simulated states of understanding* (SSOU): *ready*, *processing*, *reporting*, *busy-no-response*, and *busy-delayed-response*. Each state is intended to produce conversational behaviors on the part of the user similar to those produced by the SOU in human-human conversations. The model prescribes the type of feedback to be provided based on communication expectations but not the specific form in which it must be provided. The states are called *simulated* to emphasize the notion that this is not a model of understanding or of some other cognitive process. The model represents the communication expectations from a user's point of view. It does not address the communication expectations that the computer could or should have in a human-computer dialogue, see [17] for a discussion of this issue.

A user interface is a collection of concurrent dialogues between the user and the computing system[1]. For each one

1.In this context, system refers to the user interface component of the software system. The rest of the software system can be doing processing in the background asynchronously with what is happening at the user interface.

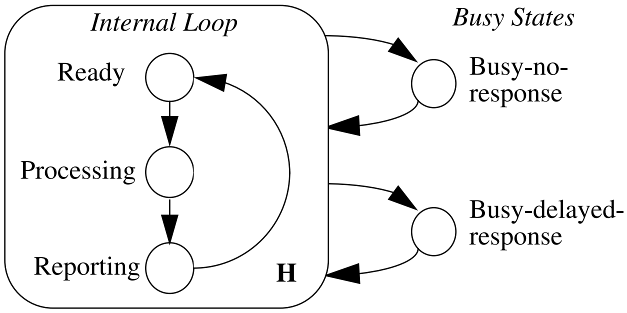

Figure 2. SSOU Feedback States

of these dialogues, the user interface *must provide* some form of evidence about the state of the dialogue. Each dialogue is *always* in one (and only one) of the states shown in Figure 2. In the performance meter example above, the application had possibly several dialogues with the user, and the performance meter had at least another one. In addition, the state of the dialogue can be indicated using more than one signal, as it is often done in human conversations. For example, a cursor change and a progress bar are used at the same time in some situations to indicate that a particular dialogue is processing a user request.

The *ready*, *processing*, and *reporting* states are known as the internal loop and represent feedback as responses to users' actions. The other two states are collectively known as the busy states and represent feedback as an indicator of system state. The notation used on Figure 2 is a modification of that presented in [12]. The internal loop has a history component. When a transition is made to one of the busy states, the internal loop "remembers" the state in which it was. When the transition is made back into the loop, it returns to the state "remembered." This is used as a short hand to avoid explicit portrayal of transitions from every node in the internal loop to both busy states and back.

Ready State

The system must provide evidence that it is in the *ready* state when it can receive and process user actions. When evidence of this state is presented, the user might initiate the next action and will expect to see some form of feedback indicating that the action has been received by the system. The behavioral role of this state is to inform the user that the system is ready to accept the next action.

Processing State

In the *processing* state, the system communicates to the user that an action was received in the *ready* state, and that something is being processed before providing the results of the action. Note that no results are displayed in this state, that is done in the *reporting* state. The behavioral role here is to inform the user that his or her action was received and it is being processed.

Sometimes, the *processing* state appears to be skipped if the results can be calculated so fast that there is no apparent delay in it. An example of this situation is dragging an object with the mouse. The object being dragged follows the mouse location continuously, giving the appearance of no time spent on the *processing* state.

On the other hand, the *processing* state sometimes requires a separate notification. For example, copying a very large number of files on the Macintosh Finder can produce feedback of processing and reporting. The system shows the *processing* state before it starts copying the files by displaying the message "Preparing to copy files…" This is a case where a separate *processing* feedback is used from the feedback used in the *reporting* state due to the length of time spent in the *processing* state.

Reporting State

In the *reporting* state, the system informs the user of the results of the action initiated in the *ready* state. Once this information is given to the user, the system is expected to go back to the *ready* state where it will be ready to accept the next user action. The behavioral role here is to inform the user of the results of a user action, thus providing closure to the action.

Busy States

Our model includes two busy states that are used to provide feedback about system state. The *busy-**no**-response* state is a state in which the system is "unaware" of any possible action the user might initiate. While in this state, all user initiated actions are ignored and lost. In the other busy state, *busy-**delayed**-response*, all user initiated actions are saved and processed later. This state has been called "type-ahead" in command line interfaces.

It is important that the busy states be signalled to the user to maintain a "sense of presence" [17] of the interface. It has been recommended that some form of "placebo" [9] is provided to ensure the user the system has not crashed. In our model we have subclassified the busy states into two separate states because of the different behavioral role each fulfills in the dialogue.

If the user is presented with evidence that the system is in the *busy-**no**-response* state, the user will opt to wait until the system returns to the *ready* state, since all actions are ignored by the system. This is the traditional busy state in most user interfaces. On the other hand, if the system is in *busy-**delayed**-response* and the user is aware of it, then the user might take advantage of the state and perform some actions even though no immediate response is provided. This has been seen in command line interfaces for years, but it also occurs in graphical interfaces.

In the *busy-**no**-response* state, the behavioral role is to communicate to the user that nothing can be done at this point. Any user actions will result in wasted effort on the user's part. In the *busy-**delayed**-response*, however, the behavioral role of the feedback is to inform the user that actions *will* be accepted but processed after a delay. In effect, it is as if the system was telling the user "I am listening, but will respond after a short delay... go ahead keep working." The analogue in human conversations is when a participant gives the other participant consent to continue talking even though no feedback will be provided. This might occur, for example, when their visual communication is interrupted and thus feedback cannot be provided using the usual mechanisms. In this case, feedback and responses might be provided only after a delay.

Unfortunately, in today's interfaces these two states normally are not signalled differently to the user. As a result, taking advantage of the "type-ahead" features is left as a "goodie" that only advanced users learn. For example, printing a document on the Macintosh using the default choices of the Print dialogue box can be done with just two keystrokes: command-P followed by the return key. Most of the time, the dialogue box will not be displayed in its entirety. This is a nice shortcut that uses a *busy-delayed-response* state, but only advanced users know about it.

Failure to represent adequately the *busy-delayed-response* state can cause undesirable side effects. Consider the case when the user interface indicates that it is in the *ready* state, but it is really in the *busy-delayed-response* state. If the user initiates the next action, the action is stored and processed after the delay. But if the context of the interface changes during the delay, the user's action might have unintended side effects. There are many examples of this in today's desktop applications. The Sun (OpenWindows) File Manager allows the user to delete files by dragging their iconic representation to a trash can. After a file has been deleted, there is a small delay in user event processing without an indication. The user sees the interface in a *ready* state and initiates the next action. During the short delay, all icons in the window are redrawn to fill the gap where the deleted file was located. This change in location of icons combined with the delay in processing causes the icon under the cursor when the event occurred to be a different one when the event is processed, resulting in the user grabbing the wrong icon.

The problem in this example is that the system indicates that it is in a *ready* state, when it really is in a *busy-delayed-response* state and the context changes before user actions are processed. The solution to this problem is simple. If a context change will occur, then the system should signal the state as *busy-no-response* and flush the event queue when coming back into the *ready* state.

Repair Behavior and Effort

In general, when a simulated state of understanding is not communicated effectively to the user, the user will enter a repair dialogue to find evidence of the correct state. The performance meter example is one example of this repair behavior. The user is not receiving enough feedback of whether the system is ready or not and thus decides to rely on the performance meter as a source of feedback. Looking at the performance meter is a repair dialogue resulting from the need for feedback when the system provides the wrong evidence, or no evidence at all.

A repair dialogue may or may not be disruptive to the user. The level of disruption of the current user's goals depends on the task at hand. What is true always, is that the user will have to spend more overall effort to accomplish his or her goals when the system is not cooperating.

Busy States

Another interesting behavioral aspect of this model is the separation of the *busy-no-response* state from the *busy-delayed-response* state. The current practice is to not signal the *busy-delayed-response* state.

When both busy states are signalled as a single state, the user will rarely take advantage of the *busy-delayed-response* state, since the information the system is providing indicates that the system is busy. Furthermore, at times the user will seek extra evidence to determine which state the system is in. That is, the user covers the collaborative gap left by the lack of appropriate feedback.

When *busy-delayed-response* has the same feedback as the *ready* state, the user will spend extra effort and possibly even obtain unwanted results because he or she will act at times when the system was not in the correct state.

If the system does provide accurate indication of the *busy-delayed-response* state, the user is more likely to take advantage of "type-ahead", or "drag-ahead" in the case of a graphical user interface. This will occur even if no feedback for the dragging action is provided during the delayed period. In this case, the user is attempting to reach the goal of the interaction without requiring the full feedback that could be provided by the system.

The busy states are a good example of where the user's repair behavior and extra effort will be spent overcoming "miscommunication" from the system. The user fills the conversational gap left by the system with extra effort exhibited in the form of repair behavior. The experiment described below studies the repair behavior and the extra effort caused by this behavior when users participate in a dialogue without good communication of the SSOU.

Related Work in the Context of Our Model

Our model incorporates many findings reported in the literature dealing with feedback and response times. The internal loop in the model includes the three states emphasized by proponents of direct manipulation interfaces [13, 14, 23]. User actions in the *ready* state require short and quick responses from the system in the form of the *reporting* state. Because most actions are short there is little need for feedback in the *processing* state.

Most response time studies [10, 11, 16, 17] are also related to this internal loop. Depending on the complexity of the action requested by the user, the time requirements for the response vary. If the user moves the mouse, he or she would expect the cursor on the screen to be moved accordingly. Such short actions, sometimes called "reflex" actions, require a response of less than 0.1 seconds [18]. In these cases, this is the timing constraint for a complete pass through the internal loop of the model. For more complex actions, for example actions that take more than 10 seconds, a progress bar should be displayed. This progress bar indicates the amount of work being done inside of either the *processing* or *reporting* state, as in the Macintosh file copying example.

A closely related model, Brennan and Hulteen's speech feedback model [3] differs from our model's structure only in that their model does not account for our *busy-delayed-response* state, possibly because it is not a common state in

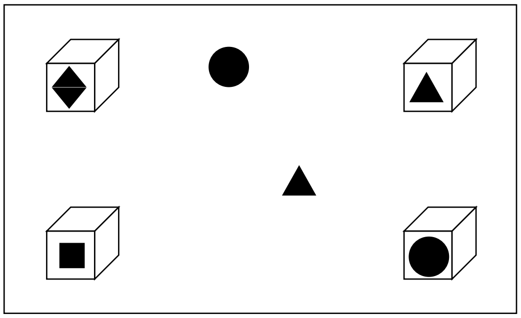

Figure 3. Screen Display in Experiment

a speech understanding system. The other states can be directly translated between the two models; some of their states are further decompositions of our states.

EXPERIMENT

We designed an experiment to explore the effect different forms of feedback have on users behavior in a direct manipulation task. Based on our model, we tried to answer three main questions: Do users engage in a repair behavior when the state of the system is not communicated correctly? Does this repair behavior amount to a significant extra effort on the user's part? And, how do users take advantage of the *busy-delayed-response* state?

Method

Thirty participants were recruited from NRL summer coop students, George Washington University computer science students and U.S. Naval Academy computer science students. All participants had comparable experience with graphical user interfaces (9 use MS-Windows, 19 use X-WIndows), and they all use the mouse on a daily basis (average 4.9 hours a day). All participants had high school degrees, 10 had completed undergraduate studies, 5 had graduate degrees.

The task to be performed by each participant involved the movement with the mouse of geometrical icons from their original location to a target location. The shapes used for the icons were circles, triangles, squares, and diamonds. The target locations were boxes with one of the shapes drawn on the front face of the box. In Figure 3, all four boxes are shown with two shapes, a circle and a triangle. Participants were asked to drag the shapes over to the corresponding box. When the object was dragged over a box, the box highlights, independent of whether it was a correct assignment or not. When the object was dropped on a box it disappeared, whether it was the right box or not.

The dependent measures used in the experiment were: number of interaction techniques (ITs), number of assignments, and number of incomplete actions. All of these measures were computed as totals and as measure per time unit (seconds). IT were all mouse-down actions, including clicks and drags. Assignments were subdivided into valid assignments and invalid assignments (assignment to the wrong box). Incomplete actions were ITs that did not result in an assignment. Time and distance moved were captured for possible future analysis but were not used in this experiment.

Procedure

The experiment consisted of three sessions of about five minutes and forty seconds each. In each session, the participant was presented with objects that appeared at a random location on the display. New objects appeared approximately every 3 seconds. The participants were instructed to assign the objects to their corresponding boxes as quickly and accurately as possible, with both criteria having equal importance.

Based on pilot data, the event rate was kept slow so that all participants would be able to complete the task. It was our desire to study the repair behavior exhibited by participants without causing frustration by making the task too difficult to complete. We were not concerned with performance effects in this study.

Three of the SSOU were used in the experiment (Table 1). In the *ready* state, all user actions were accepted and processed as soon as they occurred. In the *busy-no-response* state all user actions were ignored. In the *busy-delayed-response* all user actions were recorded but processed after a delay of approximately 2 seconds. User actions were played back after the delay, exactly as they had been performed.

State	User Action	System Response
Ready	Accepted	Yes, without a delay
Busy-No-Response	Ignored	None
Busy-delayed-response	Accepted	Yes, after 2 second delay

Table 1: SSOU in Experiment

Each session started in the *ready* state, then went to the *busy-delayed-response* state, back to *ready*, next *busy-no-response*, and then the whole cycle was repeated. Each phase was approximately 12 seconds long and participants were presented with approximately 4 objects per phase. There were a total of 29 phases, with the first one and the last one being *ready* phases, yielding a total of 7 *busy-delayed-response*, 7 *busy-no-response*, and 15 *ready* phases.

As an indication of feedback, we chose to use mouse cursor changes because this is a common use of mouse cursors in all the graphical user interfaces with which our subjects had familiarity. Treatments differed only in the type of feedback provided to indicate the state of processing. The user interface was always in one of the three states shown in Table 1. Indication of the state was given by changing the mouse cursor, as shown in Table 2. The watch in treatment B under the busy states was not reused in treatment C to avoid transfer effects in the recognition of the cursor.

Each participant was assigned randomly to one of three groups. Each group was presented with the three different treatments, counter-balanced according to a latin square design [4]. All participants received written instructions explaining the task to be performed. A three minute practice

Treat-ment	Ready	Busy-No-Response	Busy-Delayed-Response
A	Arrow	Arrow	Arrow
B	Arrow	Watch	Watch
C	Arrow	Stop Sign	Hour Glass

Table 2: Three Treatments in Experiment

session was given, during which participants saw only the *ready* state. All user and system actions were recorded for analysis. At the end of all three treatments, the participants were given a questionnaire to obtain demographic information (e.g. age, education, etc.).

To avoid biasing the participants towards a specific behavior, participants were not told of the busy states that were to occur in the three experimental conditions. They neither saw these states in the practice session nor did they know the different cursors associated with the different treatments.

RESULTS AND DISCUSSION

Performance in the task was not significantly different over any of the independent variables. This was expected, as explained earlier, because we maintained a relatively slow event rate. On the average, participants assigned 96.8% of the objects they were presented. The only significant performance-related result was the number of incorrectly assigned objects (objects assigned to the wrong box). The mean number of incorrect assignments for all participants was only 0.32 objects per session, yet there was an interaction effect between session and state of processing, $F(2, 139)= 3.89$, $p=0.0227$. In the ready states of the first session, participants incorrectly assigned more objects than in the ready states of the second session, $F(1)=8.08$, $p=0.0051$.

Repair Behavior

Our model suggests that if no information about state is provided to the user when the system is in *busy-delayed-response* or *busy-no-response*, then the user will perform extra actions to elicit the state of the system. These extra actions in our experiment are in the form of incomplete actions, clicks and drags that do not result in assignment of objects. Treatment A did not provide any feedback for the two busy states. As a result, treatment A should produce some form of repair behavior from the user.

Treatment ($F(2,224)=31.89$, $p=0.0001$) and state ($F(2,224)= 78.03$, $p=0.0001$) were significant for the number of incomplete actions per second (Figure 4), and the interaction effect between these two factors was also significant, $F(4, 224)=12.31$, $p=0.0001$. There were almost no incomplete actions in the ready phases for all three treatments (means were < 0.0001 for all three treatments). Treatment A produced more incomplete actions per second than treatment B

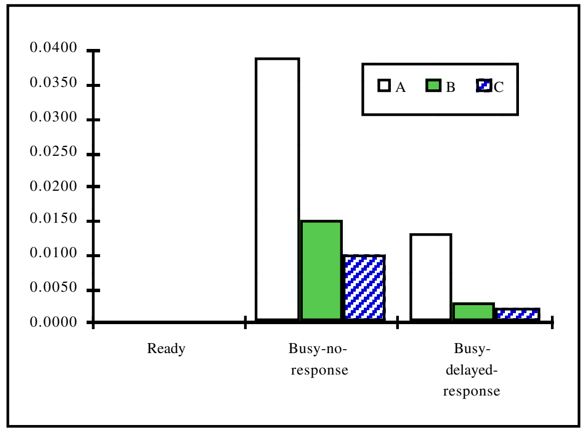

Figure 4. Incomplete Actions per Second

and C in the *busy-no-response* state, $F(1)=94.56$, $p=0.0001$. Treatment A also had significantly more incomplete actions per second in the *busy-delayed-response* state, $F(1)=15.64$, $p=0.0001$.

A very similar effect is found in total number of interaction techniques (IT) per seconds (Figure 5). This total includes incomplete actions and all the assignments[1] performed. Treatment ($F(2,224)=47.30$, $p=0.0001$) and state ($F(2,224)= 69.74$, $p=0.0001$) had significant effects on total IT, and there was an interaction effect between these two factors, $F(4, 224)=14.67$, $p=0.0001$. There was no significant difference among all treatments in the ready state. However, there was a significant difference in the *busy-delayed-response* state caused by treatment. Treatment A produced significantly more ITs than the other two treatments, $F(1)=34.37$, $p=0.0001$. Treatment B had fewer ITs than treatment C, but the difference was not significant[2].

These differences in incomplete actions and interaction techniques provide significant evidence that participants who are not given correct feedback of the state of the system engage in repair behavior. Treatment A produced the largest amount of repair behavior in the two busy states.

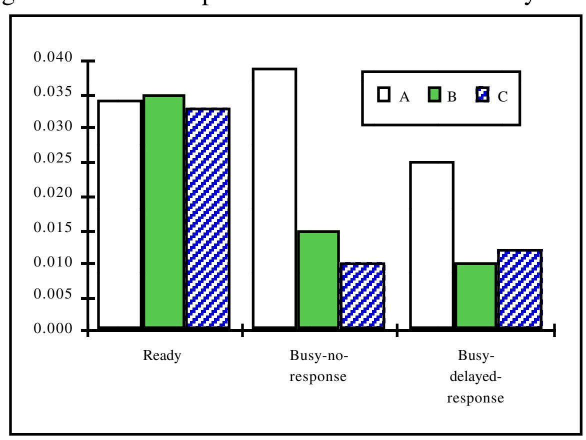

Figure 5. Interaction Techniques per Second

1.Assignments were not possible in the *busy-no-response* state.

2.The number of ITs in *busy-no-response* state is the same as the number of incomplete actions since no assignments were possible.

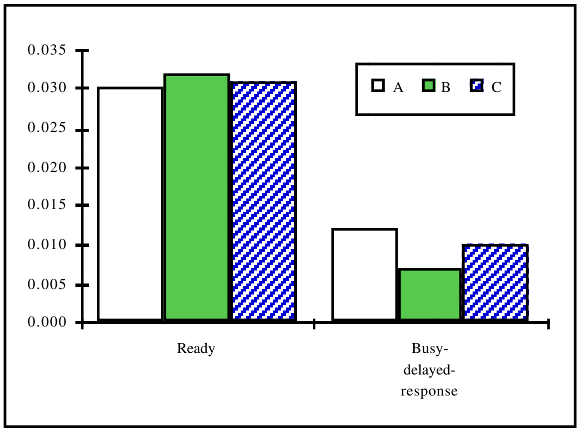

Figure 6. Total Assignments per Second

Participants in this group received no indication from the system that the state had changed. They not only had to figure out what was happening once they clicked for the first time in that state, but also had to continue clicking to determine when either busy state was over. Anyone who has ever faced a situation in which the computer does not respond to their actions understands this basic behavior.

Behavior in *busy-delayed-response*

We found evidence that users can take advantage of the *busy-delayed-response* state, once they understood that the system is responding after a delay. The participants had no a priori knowledge of the busy states or the meaning of the cursors. So in treatment A they had to figure out what was happening in the busy states without help from the system. Treatment B provided the same feedback, the wrist-watch, for both busy states. Treatment C provided differing feedback, but its meaning was not specified ahead of time. So, subjects in Treatment C also had to do some experimentation to understand the states.

State of processing had a significant effect $F(1,139)= 1256.22$, $p=0.0001$ on the number of assignments per seconds done (Figure 6). There were significantly more assignments per second in the *ready* state than in the *busy-delayed-response* state. We also found an interaction effect between state and treatment $F(2,139)=11.59$, $p=0.0001$. Treatment A had more assignments per second in the *busy-delayed-response* state than C, $F(1)=4.09$, $p=0.0450$, and B, $F(1)=22.63$, $p=0.0001$. Treatment B had significantly fewer assignments per seconds than treatment C, $F=7.475$, $p= 0.0071$. There was no significant difference due to treatment in the ready state. Session had a significant effect, $F=3.100$, $p=0.0482$; there were more assignments in the first session than in the other two, $F(1)=6.11$, $p=0.0146$. This represents a slight learning effect, but there was no interaction with any of the other variables.

Participants in all treatments were successful assigning objects in all states. Performance in the *busy-delayed-response* state was slower than in the *ready* state. Nevertheless users were able to perform assignments.

Surprisingly, participants in treatment A assigned the highest number of objects in the *busy-delayed-response* state. Perhaps this is merely an artifact of the smaller number of assignments for treatment A in the *ready* state. It is also possible that subjects in Treatment A had more trouble identifying the beginning of each new ready state and therefore did not make full use of it, hence making fewer overall assignments in the *ready* state.

Although the difference was not statistically significant, participants in treatment B did assign fewer objects during the *busy-delayed-response* state than those in treatment C. A possible explanation is that participants in treatment B assumed a more passive role in their *busy-delayed-response* state because they received the same feedback for both busy states. They could be interpreting both states as *busy-no-response*. Further data would be needed to fully explain this result.

CONCLUSIONS

We have taken a model of human conversation and reformulated some of the principles to cover human-computer interaction feedback. This is similar to the approach described in [2, 9, 17, 19]. Here, a theory of collaborative conversations served as the basis for the definition of states of feedback, with each representing different communication expectations users have when interacting with a computer.

The experiment showed that failure to properly identify a SSOU produces repair behavior by users, and that this repair behavior can be a significant amount of their effort. The study also shows that a different behavior is associated with each of the two busy states. It provides evidence that if a delayed state is identified, users do indeed take advantage of this “type-ahead” state.

The model presented, coupled with the feedback timings reported in the literature, gives the user interface designer the feedback requirements for the design of new interaction techniques and dialogues. More importantly, our model does so in a style-independent manner, so it is not specific to one platform or one style of interaction. As is common in our field, adherence to the principles of this model does not guarantee a good interface. But based on the empirical evidence and on our observations of commercial applications, violation of the principles embodied in the model does produce disruption on the human-computer dialogue as evidenced by the repair behavior. In this light, the model provides feedback design guidelines for interaction techniques and human-computer dialogues.

The model prescribes the feedback states for each dialogue in a human-computer interface. But, each user interface can have several human-computer dialogues, with each dialogue possibly having multiple cues communicating state information. The user integrates several cues into a single communicative event to identify the SSOU of the system. The model presented here does not address how this combination is done. It seems that some hierarchical combination of cues is done, but more work is needed to extend the model in that direction.

Finally, the goal of our research is to identify principles from human conversations that are desirable in human-computer dialogues [21, 22]. The feedback model presented here provides a behavioral description of direct manipulation interactions and their feedback requirements. This model is the lower level of a human-computer dialogue framework, currently under development, that studies dialogue at the feedback level and at the turn-taking level.

ACKNOWLEDGMENTS

We would like to thank Jim Ballas, Astrid Schmidt-Nielsen, and Greg Trafton for their many helpful comments about the design of this experiment. We also would like to thank the Computer Science Department at the U. S. Naval Academy and Rudy Darken at NRL, for allowing us to run most of the study at their facilities. The GWUHCI research group provided many helpful ideas in the design and the analysis of the data. This research was funded in part by the Economic Development Administration of the Government of Puerto Rico, and by NRL.

REFERENCES

1. Apple Computer Inc. *Human Interface Guidelines: The Apple Desktop Interface*. Addison-Wesley Publishing Company, Inc., Reading, Masachusetts, 1987.
2. Brennan, S.E., Conversation as Direct Manipulation: An iconoclastic view, in *The Art of Human-Computer Interface Design*, B. Laurel, Editor. 1990, Addison-Wesley Publishing Company, Inc.: Reading, Massachusetts.
3. Brennan, S.E. and Hulteen, E.A. Interaction and Feedback in a Spoken Language System, in *AAAI-93 Fall Symposium on Human-Computer Collaboration: Reconciling Theory, Synthesizing Practice*, (1993), AAAI Technical Report FS93-05.
4. Bruning, J.L. and Kintz, B.L. *Computational Handbook of Statistics*. Scott, Foresman and Company, Glenview, Illinois, 1987.
5. Clark, H.H. and Brennan, S.E., Grounding in Communication, in *Shared Cognition: Thinking as Social Practice*, J. Levine, L.B. Resnick, and S.D. Behrend, Editor. 1991, APA Books: Washington, D. C.
6. Clark, H.H. and Schaefer, E.F. Collaborating on contributions to conversations. *Language and Cognitive Processes, 2, 1* (1987), pp. 19-41.
7. Clark, H.H. and Wilkes-Gibbs, D. Referring as a collaborative process. *Cognition, 11,* (1986), pp. 1-39.
8. Foley, J.D. and Dam, A.v. *Fundamentals of Interactive Computer Graphics*. Addison-Wesley System Programming Series, ed. I.E. Board. Addison-Wesley Publishing Company, Reading, Massachusetts, 1982.
9. Foley, J.D. and Wallace, V.L. The Art of Natural Graphic Man-Machine Conversation. *Proceedings of the IEEE, 62, 4* (1974), pp. 462-471.
10. Gaines, B.R. The technology of interaction- dialogue programming rules. *IJMMS, 14,* (1981), pp. 133-150.
11. Gallaway, G.R. Response Times To User Activities in Interactive Man/Machine Computer Systems, in *Proceedings of the Human Factors Society-25th Annual Meeting*, (1981), pp. 754-758.
12. Harel, D. On Visual Formalisms. *CACM, 31, 5* (1988), pp. 514-530.
13. Hutchins, E.L., Hollan, J.D., and Norman, D.A., Direct Manipulation Interfaces, in *User Centered System Design: New Perspectives on Human-Computer Interaction*, D.A. Norman and S.W. Draper, Editor. 1986, Lawrence Erlbaum Associates: Hillsdale, NJ.
14. Jacob, R.J.K. Direct Manipulation, in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, (Atlanta, GA, 1986), IEEE, pp. 384-388.
15. Lee, K.-F. The Conversational Computer: An Apple Perspective, in *Proceedings of Eurospeech*, (1993), pp. 1377-1384.
16. Miller, R.B. Response time in man-computer conversational transactions, in *Proceedings of Fall Joint Computer Conference*, (1968), pp. 267-277.
17. Nickerson, R.S. On Conversational Interaction with Computers, in *User Oriented Design of Interactive Graphics Systems: Proceedings of the ACM SIGGRAPH Workshop.*, (1976), ACM Press, pp. 681-683.
18. Nielsen, J. *Usability Engineering*. AP Professional, Cambridge, MA, 1993.
19. Payne, S.J. Looking HCI in the I, in *Human-Computer Interaction - INTERACT '90*, (1990), Elsevier Science Publishers B.V., pp. 185-191.
20. Payne, S.J. Display-based action at the user interface. *IJMMS, 35,* (1991), pp. 275-289.
21. Pérez, M.A. Conversational Dialogue in Graphical User Interfaces: Interaction Technique Feedback and Dialogue Structure, in *Proceedings Companion of the ACM CHI'95 Conference on Human Factors in Computing Systems*, (Denver, Colorado, 1995), Addison-Wesley, pp. 71-72.
22. Pérez, M.A. and Sibert, J.L. Focus on Graphical User Interfaces, in *Proceedings of the International Workshop on Intelligent User Interfaces*, (Orlando, Florida, 1993), ACM Press, pp. 255-257.
23. Shneiderman, B. The future of interactive systems and the emergence of direct manipulation. *BIT, 1,* (1982), pp. 237-256.
24. Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison-Wesley Publishing Co., Reading, Masachusetts, 1987.
25. Simes, D.K. and Sirsky, P.A., Human Factors: An Exploration of the Psychology of Human-Computer Dialogues, in *Advances in Human-Computer Interaction*, H.R. Hartson, Editor. 1988, Ablex Publishing Corporation: Norwood, New Jersey.
26. Teal, S.L. and Rudnicky, A.I. A Performance Model of System Delay and User Strategy Selection, in *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems*, (Monterey, California, 1992), Addison-Wesley, pp. 295-305.