

# The Strange Absence of Abstraction Levels in Designing HCI

Dov Te'eni<sup>(✉)</sup>

Tel Aviv University, 61390 Tel Aviv, Israel  
teeni@post.tau.ac.il

**Abstract.** People process and communicate information at multiple levels of abstraction when reading, talking, solving problems, designing and interacting with computers. For example, in reading an article, actors may focus on a letter, a word, a clause, a sentence or a paragraph. At any moment, they focus on a particular level of abstraction, do something, and, under certain conditions, move back and forth to other levels until the actors achieve their goal. Not moving between levels of abstraction when necessary, decreases performance. It follows that human-computer interaction should be designed accordingly, yet there is hardly any explicit mention of abstraction levels in studies or guidelines of designing HCI. In this talk, I propose a method for incorporating abstraction levels in the design of HCI as a critical dimension of designing adaptive HCI. The talk demonstrates the ideas with examples of HCI for supporting online reading and group problem solving.

**Keywords:** HCI design · Feedback · Levels of abstraction

## 1 Introduction

The notion of levels of abstraction (LoA) has been used in numerous contexts of human information processing (Vallacher and Wegner 1987). For example, in reading this article, readers can process information at different levels such as word, sentence and paragraph. We restrict the discussion to goal directed behavior and assume that the readers have a goal they wish to accomplish, e.g., understand the article's main message or edit the article, and that they choose to concentrate on the LoA corresponding to their goal. For example, readers wishing to understand the logic of the arguments, may read the article at the level of a sentence, i.e., attempt to understand the information given in each logical step, occasionally skipping an unknown word if it does not interfere with grasping the general message of the sentence. In comparison, readers wishing to understand the article's theory and method, may read it at the level of a paragraph, i.e., attempt to grasp the message of the entire paragraph by glossing through the paragraph, seeking signs of the general message with perhaps greater attention to the beginning and ending of the paragraph or to anything in bold or italics. (Of course, we read articles in many other ways, e.g., read only its title.)

The readers' choice of a focal level of attention depends on their goal and the situation in which they pursue their goal, e.g., understand the main message of the article when

in a hurry or edit the article carefully before final approval to print. Regardless of the initial choice of a focal abstraction level, readers usually move from one LoA to another during a session of reading, e.g., they oscillate between the sentence level and the paragraph level, and at any moment, they attend to a particular focal LoA (Vallacher and Wegner 1987). These moves across levels of abstraction can be viewed as an adaptive behavior that is effective when the moves result in achieving the behavioral goals successfully.

How can designs of online articles take into account LoA? Unlike a printed article, an online article (like an e-book) affords the adaptation of the human-computer interaction (HCI) to the dynamic reading behavior of the user. The HCI can be adapted to fit the abstraction level of the user's choice or to fit an optimal abstraction level determined by the computer. In reviewing the HCI literature on designing e-books and readers, I could not find explicit consideration of LoA. Nor could I find explicit reference to LoA in the practice of designing e-books. Indeed, I found very little on LoA in design of information systems more generally, albeit, I found some limited examples of practice in certain applications that support *de facto* work at different LoA without explaining its rationale. This state of affairs is unfortunate because we are robbing ourselves the opportunity to improve future designs by considering LoA.

In the reading example, assuming that such moves between reading modes are advantageous, there are at least three broad design implications. One implication is to adapt the system so that it best serves each reading mode in turn. For instance, when scanning the article in a high-level reading mode, the reader tends to ignore less important pieces of information according to a principle called 'visual hierarchy' (Djamasbi et al. 2011), implying a design that lets the unimportant pieces almost disappear into the background. A commonly used technique to emphasize parts of the text and downplay surrounding texts is to present the page in a fisheye view. A second implication of the LoA implication is the need to design systems that support an easy transition between reading modes. For example, the system should signal clearly how to move from the current LoA to a higher or lower level and signal the arrival at the new LoA. A third, more complex, design implication of the two LoA is to prompt the user to move effectively between modes. To do so, the system has to rely on knowledge about the effectiveness of each approach for a given reading task. For instance, knowing the reader is looking for a practical solution, the system might suggest a move to a detailed-level reading when the reader gets closer to the appropriate section in the article. None of these design implications are widely implemented in online readers even though they are technically feasible, today more than ever before.

The idea of LoA is not new in systems design. Rasmussen (1986) advocated a means-ends approach to task analysis as a basis for designing systems. In more concrete and spatial aspects of HCI (i.e., working at lower levels of abstraction), the potential of interactive systems is clearer and more often utilized. An explicit consideration of LoA that I found appears in the treatment of visualization (Christoff et al. 2009). For example, interactive maps let users zoom in to a more detailed map or out to a less detailed map, which is akin to moves between LoA in more conceptual tasks. By the same token, we could expect to see systems designed to consider LoA in the treatment of abstract and symbolic tasks such as reading an article or solving a problem.

Moreover, advanced technologies are already available that can detect intentions to move to higher or lower LoA (Christoff et al. 2009) or track eye movements that can identify the user's focus of attention. Such advances in the technology will allow automatic adaptation to the desired LoA. It follows that there is a need for human-computer interaction to consider LoA in its design and that such designs are feasible technologically, and yet there is no systematic way to encourage, or even ensure, that designers incorporate LoA in their design of HCI. As these more advanced technologies become more prevalent and readily available, the desirability and feasibility will grow.

The paucity or even absence of LoA in the research and practice of HCI is puzzling. It may be a lack of awareness or the lack of methods to analyze and design LoA in HCI. Whatever the reason, not designing for LoA is a missed opportunity to improve the HCI. This talk proposes a systematic way of incorporating the idea of LoA into the design of HCI in order to achieve designs that capitalize on the idea, as demonstrated above. It begins by extending the idea of LoA in interactive systems and tying it to the design of feedback, which constitutes an important aspect of designing interactive systems. Thus, taking a critical component of system design, namely feedback, I show how designing for LoA can improve user performance by producing a better HCI. I then go beyond the idea of the LoA to suggest that the same line of thinking can be used to support the design of HCI that caters for adaptive behavior more generally.

## 2 Levels of Abstraction (LoA) and Feedback in HCI

### 2.1 LoA

An early discussion of LoA in the IS literature examined the process of solving problems in data modeling (Srinivasan and Te'eni 1995). To build an entity-relationship diagram (ERD) of a system, data modelers engage in planning, building, testing and refining, and do so at different LoA. The LoA include, from low to high LoA, properties of an entity, entities, clusters of entities and composites (e.g., a vehicle is a cluster of car and truck). This approach has been used in several studies since, e.g., Rittgen (2007) used it to study negotiation in modeling. Data modeling involves several activities such as planning and testing. What is most relevant however to our discussion is that while engaged in modeling an entity-relationship diagram of a system, data modelers adapt their modeling behavior by moving their focus of attention up and down LoA. Figure 1 shows the transitions of a participant between data-related levels of abstraction engaged in planning and testing their data model.

The figure was established based on a protocol of a think-aloud session intended to solve a data-modeling exercise to build an ERD that could enable users to answer certain queries. In their study, Srinivasan and Te'eni (1995) show that different LoA are needed to solve the problem and that moving between LoA at certain points of time enhances performance. Shifting levels can be seen as adapting to certain situations. Adaptive behavior involves working on one level, say entities, and climbing to a higher level of clusters when progress loses direction or dropping to a lower level of properties when modeling or testing fails.

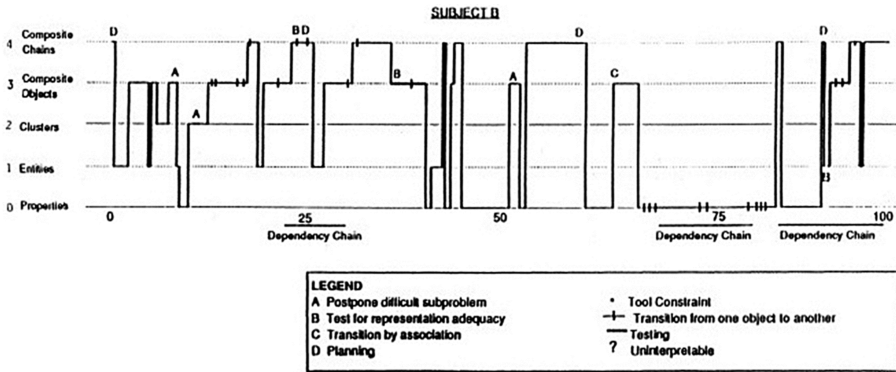


Fig. 1. Adaptive behavior in data modeling (from Srinivasan and Te'eni 1995)

Srinivasan and Te'eni (1995) explained these transitions between LoA on the basis of early work on problem solving introduced by Duncker (1945). One shift is to a higher level of abstraction, which is expected to occur when the problem solver gets lost in detail and needs the broader context to understand the details or to decide how to proceed (Duncker 1945). Another shift is to a lower level of abstraction, which is expected to occur when the problem solver feels that the higher level can no longer guide him or her on how to proceed or how to understand the problem. Importantly, Srinivasan and Te'eni (1995) showed that the transitions between LoA have consequences on performance.

From a design point of view, designs of systems that support modeling should, at the minimum, the patterns of adaptive behavior shown in Fig. 1. In this respect, these patterns represent the knowledge base required in designing for adaptive behavior in modeling.

Having identified the transitions across LoA and having characterized these shifts as adaptive user behavior, we ask what, if at all, are the consequences of certain patterns of transition. In other words, why adapt and what consequences does adaptive versus non-adaptive behavior have. Knowledge of data modeling suggests that certain patterns are more effective than others in problem solving. For instance, working mostly at lower levels of abstraction was less effective than working at higher levels, although working only at higher levels of abstraction was not effective. An effective pattern was to work at the level of entities, but from time to time to climb to higher levels for short periods. There are other effective patterns, which may depend on the user's level of expertise (Srinivasan and Te'eni 1995). In any event, in this example and on the basis of knowledge about the specific phenomenon of data modeling, we can determine the relationship between patterns of LoA transitions and consequences. In other cases, we may be able to define LoA but may not be able to determine the consequences of alternative patterns.

Another application of LoA in design looked at the design of email (Te'eni and Sani-Kuperberg 2005). This work built on the notion of a focal LoA developed by Vallacher and Wegner (1987) in their theory of action identification. The theory of action identification is rooted in human information processing and claims that people represent and communicate actions at multiple levels of abstraction, and at any one moment, one of these levels may act as their focal level (Berger 1988). People tend to remain at the focal

level, but shift their attention when complexity increases and breakdowns in understanding occur (Vallacher and Wegner 1987). People may shift to lower levels of abstraction when they feel that the higher level can no longer guide them as to how they should proceed or understand the message, or shift to higher levels when they get lost in detail and need a broader context for guidance. It follows that emails could be designed at multiple levels so that receivers of the message could work effectively at their current focal LoA, e.g., focus on the highest level of the message but revert to lower levels only when needed. For example, users see a message at a high level of abstraction and only when they find themselves unable to continue reading because a sentence is unclear, a more detailed presentation of the email appears (Te'eni and Sani-Kuperberg 2005). In other words, at any moment of work, the design must support the focal LoA and enable an easy transition to other LoA when necessary.

## 2.2 Feedback

So far, we have looked at LoA as an element of dynamic behavior and its effect on user performance. This section discusses feedback in order to examine how LoA should be taken into account when designing feedback. Feedback is arguably one of the most important aspects of HCI designs, particularly in systems that support dynamic behavior such as decision support systems (Te'eni 1992). My working assumption is that systems must be adaptive to support adaptive behavior such as moving across LoA.

Feedback has been studied extensively for years in the fields of human behavior (Annett 1969), management (Ilgen et al. 1969), decision making (Sterman 1989). In the context of human-computer interaction to support management or decision tasks, feedback can be conceived as communication between the computer and the user or as computer-mediated communication between users. As we shall see later on, both conceptualizations are relevant to design of systems that support joint or group decision-making. In either case, the first step of designing feedback is to articulate its functions and the means by which enable these functions such as choice of media and format. A primary function of feedback is to support control over the user's goal directed behavior by monitoring the direction and motivation governing the behavior.

Following the view of feedback as an element of the communication between the user and the computer, I define *feedback* as information to the user, about the user's goal-directed behavior and in response to the user's interaction with the computer system (adapted from Ilgen et al. 1979). The function of feedback is directional or motivational. Directional relates to behavioral outcomes that should be accomplished, while motivational relates to outcomes associated with rewards. Practically, feedback may affect behavior in both ways at the same time by directing behavior and incenting or reinforcing behavior (Ilgen et al. 1979). A third possible function of feedback is to support learning, which is particularly important when the relationships between the factors contributing to outcomes are complex and there is a need to develop and understanding or skill to accomplish the task (Te'eni 1992). Learning usually relies on feedback that relates not only to the outcome but also to the process leading up to the outcome, in which case it is labeled process feedback rather than outcome feedback. In the design examples discussed here, we will design outcome feedback in both directing and motivating or

reinforcing behavior, as well as process feedback that explains relationships. (Process feedback, sometimes labeled cognitive feedback, relates to the process leading up to the outcome.)

However feedback is categorized or whatever function it plays, feedback is always tied to some goal the user wishes to pursue by behaving in certain ways. Goals can generally be decomposed into sub goals, each sub goal pursued with a behavior that the user attempts to control, and appropriate feedback can support the control. Furthermore, the goal decomposition defines the hierarchy of the LoA, and each LoA may therefore carry different forms of feedback.

For example, the overall goal of creating an ER diagram (ERD) can be decomposed to several sub goals such as determine an entity's properties, check that all properties are allocated to entities, and determine the relationship between two entities. Feedback at the highest level should provide information about how the ERD is progressing toward the goal of a complete and semantically correct diagram. A second feedback, feedback at a lower level, may provide for each entity a list of defined and undefined properties and present the fraction of properties already defined; this is feedback towards the goal of completing the determination of the entity's attributes. A third form of feedback could be generated while in the process of building a relationship between two entities. Say the user is in the process of selecting the type of relationship from a pull-down menu a candidate for the relationship (e.g., a one-to-many relationship), useful feedback would indicate the correctness of the relationship selected (e.g., that the relationship is infeasible given the specified properties). Each of these three forms of feedback can be associated with a particular LoA. The next section explains how the perspective of LoA informs the design of feedback.

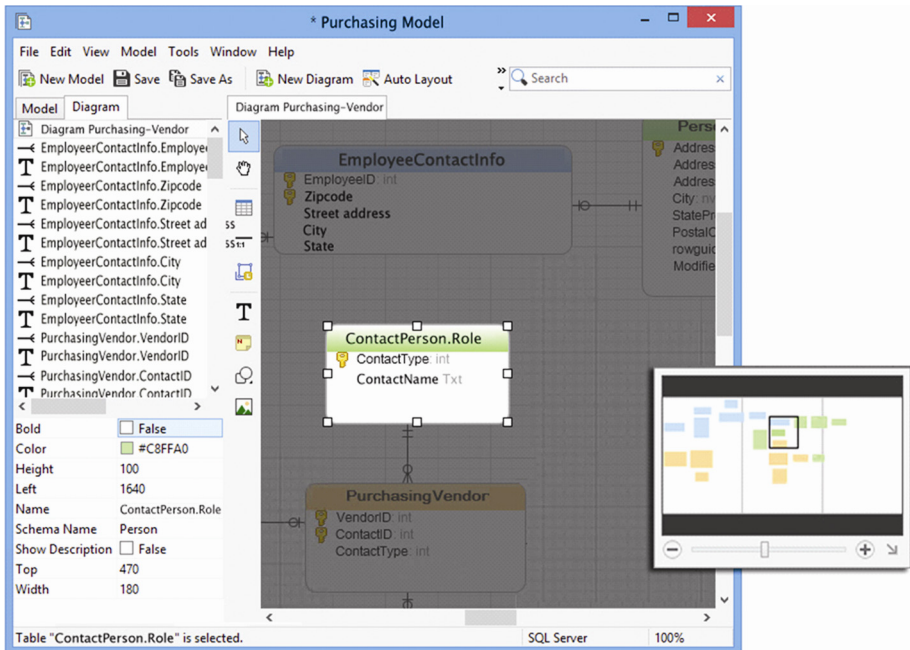
### 3 Design for LoA

To recapitulate, the LOA perspective posits that people have a focal LoA at any moment of their work and that they move back and forth between LoA when necessary. Therefore, the design of feedback (and of other aspects of HCI) should adapt the design to support the focal LoA. As the focal LoA changes during the interaction, it will also be necessary to support an easy transition between levels when the user moves on her own initiative or manage and initiate transitions automatically. Using the data modeling example, we use

#### 3.1 Fit the System to the State to Which the User Moved

Figure 2 depicts a screen that supports building and testing an ERD at different LoA, as described above. The rectangle describing the entity `Person.ContactType` fits the task the user is expected to perform, namely to determine and check the properties of the entity, part of which are detailed in the lower left corner (e.g. Bold is set to False). A better fit could be to move the table of properties from the lower left corner closer to the central rectangle so that it takes minimal effort to move between the two. In fact, a clever design might be able to integrate the two, in order to bring the manipulations on the

entity's properties as directly as possible, e.g., when the cursor hovers over the (green) colour of the rectangle, a colour palette appears inviting the user to choose another colour.



**Fig. 2.** A screen for working at the properties level of abstraction (Color figure online)

Designing for habitual behavior differs from designing for novel behavior that requires mindful planning and controlling. When fitting the human-computer interface to the user's expected behavior, the designer's goal is to minimize the user's effort in performing the task correctly. When fitting the human-computer interface in novel situations, the designer's goal, in addition to minimal effort and accuracy, is to support the generating of new behavior and controlling unpracticed behaviors. In the case of setting or manipulating an object's properties, behavior is assumed to be habitual (unless the user is in training). The screen is designed to allow easy manipulations of data, immediate feedback (e.g., setting the color immediately shows the color to be displayed during use under the anticipated conditions of use), and easy and therefore more accurate detection of properties and their values.

When working at a higher LoA, namely at the level of entities, the user is expected to plan the relationships between the entities. The small map of entities, when expanded, would become the centerpiece of the screen. Its graphic presentation is a better (cognitive) fit than, say, a table of relationships, because it lends itself most immediately to the way people represent relationships in their mind (Vessey and Galletta 1991).



### 3.2 Design Systems to Support Transitions Between LoA

The most obvious design implication, often ignored in practice, is to enable easy transitions between LoA. This would mean, for example, an easy transition between entities and clusters of entities. If we know that users tend to move from one level to another on a dimension, the system should be designed to support these movements, even if the consequences of transitions are not known (at least if it is not known to be detrimental). Easy transitions between levels of abstraction mean both easily operating the system to reveal and move the focus of attention to another level and supporting the transition cognitively (Sun 2012). The move from one level to another usually means breaking away from the habitual behavior into which the user has settled and requiring the user actively think how to proceed, and this requires some new and forceful condition or some external trigger (Louis and Sutton 1991).

Technically effecting a transition to another level usually involves some form of direct manipulation, like a single click to zoom in or an option to hover over an entity, reveal its properties and move to one of them. Cognitively supporting the transition between levels includes at least two types of support: underscoring the new state as the current focus of attention, and maintaining the source as context when arriving at the target. In other words, the user often needs to realize that the focus has shifted to a new level of properties. And at the same, he or she needs to take time to see the level of entity as the context for working on the properties. For example, in Fig. 2, the user has moved from working on entities to focusing on the properties of a particular entity labeled 'Person.ContactType'. The small map of entities (the higher level of abstraction from which the user moved to the current focus) is left on the screen to present the higher level as the context for the current focus. At any time, the user can go back to the higher level by clicking on the small map.

### 3.3 Design Systems that Guide Advantageous Behavior

The last of the five steps of the procedure for designing adaptive behavior systems requires knowledge of performance measures in the particular domain of the user's work. Furthermore, this step is only feasible if there is a convincing argument for affecting consequences advantageously by manipulating the human-computer interaction (step #2 above). The research quoted above argues that certain patterns of transition are more effective than others. For example, remaining too long at the lowest levels of abstraction without occasionally taking a more comprehensive view by climbing to a higher level of abstraction will cause people to make errors. The system can detect fairly long periods of working on properties by monitoring the user's manual inputs to the system (in other cases, the system could monitor the user's gaze). The system can then alert, suggest, or force corrective action like moving to a higher level. In other cases, it may be important to consider not only the proportionate time spent at different levels, but also the sequence of activities.

Technologies that monitor users' behavior and conditions are becoming widespread in some areas like health and safety at work, but will most likely spread to many other domains of life. The nearly constant accessibility to devices like the cell phone, online



watch and wearable devices in general, as well as knowledge about what is effective in which conditions makes it feasible to guide advantageous behavior. A simple example is RunKeeper's (an App to plan and monitor jogging activities) real time health graph, which could easily be supplemented with alerts of when you should accelerate to meet your running goal, or slow down to maintain your health. Similarly, as well as signaling when the user should move to another level of abstraction, a modeling system could motivate the user with graphical depictions of successful patterns vis. a vis. her own actual pattern.

## 4 Design of Feedback in Group Decision-Making

### 4.1 Task, LoA and Feedback

Feedback in computerized systems that support joint (or group) decision-making can be conceived within two contexts of communication: communication between the computer and the user (as was the feedback described in the data-modeling example above) and communication between users mediated by the system. This section discusses feedback in the context of a personnel-screening task in which a team of recruiters wishes to screen candidates. The recruiters initially work alone to form their individual judgments of the candidates but eventually are exposed to the judgments of their fellow recruiters. As a final step, the recruiters work jointly to negotiate a collective ranking. As demonstrated below, communication between recruiters provides opportunities for interpersonal feedback between colleagues in addition to feedback generated by the computer as part of the HCI.

Sengupta and Te'eni (1993) experimented with this setting to test the impact of feedback on the decision maker's cognitive control. They created teams of three recruiters that ranked applicants according to structured profiles composed of three attributes: work experience, test scores, and education. To avoid biased judgments, the recruiters did not receive other information, such as photos, gender or age. Each of the three attributes was presented to the recruiters as an integer evaluation (on a 1–9 scale). The recruiters (users) worked individually on desktop computers but were linked through a network so that they received information about their colleague's judgments as they progressed with their own judgments.

The recruiter's first goal was to evaluate ten candidates. The evaluations from the three recruiters would later serve as a basis for a collective ranking of the ten candidates. The user (recruiter) worked at two broad LoA: setting the weights for the three attributes, which applies to the entire set of candidates, and evaluating each candidate. The system supports both levels with appropriate feedback. The user can begin by setting the weights and receiving feedback that visually reflects the relative importance of the three attributes and compares the current weights to previous weights and to weights set by others. The user can then inspect each applicant's profile and rate the applicant overall (also on a 1–9 scale), receiving feedback computed according to a theoretical model of social judgment (Hammond et al. 1980). At the user's discretion, feedback from fellow recruiters showing their ratings can be shown side by side with the user's rating so that the user has the opportunity to adapt her or his own behavior by changing ratings.

4.2 Fitting the System to the Focal LoA

Figure 3 shows the screen for working alone (the user is John) at both LoA: determining the overall strategy (setting the weights) and rating an individual candidate. How does the screen design adapt to a change in the user's focal LoA? When the user moves to setting weights, the working area that includes the relevant feedback is highlighted and the rest of screen is dimmed. The relevant feedback includes the newly inputted weights in comparison to previous weights as well as cognitive (process) feedback showing visually on the horizontal bar chart, how each bar stretches or shrinks as each weight is changed. Moreover, the consistency of applying the strategy is also shown. The theory-based index of consistency is feedback about the implications of the user's action in setting the weights. Thus, the entire set of feedback signals are designed to hold the user's attention to the focal LoA.



Fig. 3. Working alone screen for ranking candidates

Similarly, when the user moves to work at the level of a single candidate, the corresponding feedback must be at the focus of attention. Say the user John is considering the values of the three attributes in the highlighted row of the fourth candidate. The values (6, 6, 1) for the work experience, test scores and education, respectively), feedback showing the predicted evaluation appears under 'computer'. In other words, after the user inputs an evaluation (3), the user's predicted evaluation (5) is feedback generated by the computer. The user can always examine the feedback shown at the higher

level to guide individual evaluation, which is essentially a move up the LoA to see the context of the lower LoA.

Once the user elects to communicate with the two other recruiters, their input and predicted values are displayed in the adjacent columns, as shown in Fig. 4. This is feedback that comes from others but is relevant to the user’s action. This feedback at the level of evaluating a single candidate. Feedback involving other recruiters is also generated for the higher LoA that includes a match between pairs of recruiters, their consistency and a comparison of their weights with to the user’s weights.



Fig. 4. Working with others in ranking candidates

4.3 Supporting Transitions Between Levels

In this case, the transitions between LoA are relatively easy because the upper and lower parts of the same screen are respectively the higher and lower LoA. The awareness of the two levels is high and the affordance of the system for shifting LoA is obvious – all

the user has to do is to move the cursor to the area of interest. In contrast, the shift (toggle) from working alone to working with others requires pressing the  $\pm$  or the buttons at the bottom of the screen. Beyond the physical motion to shift levels, the two levels are syntactically similar, using the same colors for attributes, as well as the order of recruiters. The formatting decisions help the user move from one level to another by minimizing cognitive effort in the move. The logical relationship between levels is further clarified by showing the change in predicted values when changing the user's weights (this happens when the user moves from the upper area towards the lower area, having changed the weights).

#### 4.4 Guiding Advantageous Behavior

The feedback provided in the experiment contributed significantly to the users' control over their judgments and improved their performance (Sengupta and Te'eni 1993). Interestingly, the one of the major drops in the user's control reflected by sharp decreases in consistency of applying strategies happens when the users first see the judgments of others and strive to adapt their own strategies in order to get closer to a joint ranking. This is when feedback is essential and most effective.

The system should be designed to monitor the user's behavior in order to trigger a move to the higher LoA when consistency drops. This can be done by flashing or resizing the consistency or match indices in the upper area.

### 5 Expansion to Adaptive Behavior More Generally

I have concentrated on examples of designing feedback for behavior that adapts by shifting across LoA. However, more generally, we increasingly need to design the human-computer interface of systems to support complex tasks, where users adapt dynamically to the task as it goes along, or to changing conditions during the interaction. There is abundant research on adaptive systems that adapt to the user's characteristics and preferences or to the conditions at the time of use (Billsus et al. 2002). There is also research on how users adapt their use of systems, e.g., using more or less features of the system (Sun 2012). Unfortunately, we lack procedures and methods to build systems that support the user's adaptive behavior when *performing a task* in the same manner we explored methods for considering transitions in LoA.

This section generalizes the discussion so far to formulate it as an approach that identifies dimensions on which users adapt their behavior, and then determines the corresponding design implications on how the system should adjust to fit the user's adaptive behavior. As noted above, technologies that supply users with physical and physiological data such as location or blood pressure to help them adapt their physical behavior are commonplace. Our concern is with mental tasks like reading or solving problems that rely on abstract feedback in order to adjust behavior. Future work will examine technical solutions based on advanced technologies for data acquisition and analytics to supply appropriate feedback but for now, I generalize our discussion assuming current technologies.

I propose a procedure of five steps to design systems that support adaptive behavior (regarding moving across LoA as one case of adaptive behavior):

1. Identify dimensions for adaptive behavior in a given activity
2. Determine the consequences of transitions between states on a dimension
3. Design systems to support transitions between states
4. Fit the system to the state to which the user moved
5. Design systems that guide advantageous behavior

For example, adaptive behavior can be transitions between a mode of automatic behavior, when things are familiar, to controlled behavior, when things become strange or unfamiliar (Louis and Sutton 1991).

Our examples have looked at how to support an individual's adaptive behavior by achieving better cognitive fit within modes and optimizing moves between modes. We can use the same approach with other aspects of supporting adaptive behavior, beyond individual cognition. For instance, nowadays modeling involves working in different modes: alone, in intimate groups and in large groups. Applying the five-step approach and relying on knowledge of the effectiveness of the different modes, will change the design of many enterprise wide systems. Another telling example is the impact of adapting information about the user (personalization) according to the user's dietary behavior with a system that shows the effect of food consumption on the user's health (Ronen and Te'eni 2013). The study compared personalized versus generalized information with respect to the users' attitudes and behavior. Subjects receiving the personalized feedback reported a more positive attitude and a greater propensity to follow the doctor's recommendations in order to improve their well-being. In other words, a good fit not only supports adaptive behavior but also shapes it by affecting attitudes.

In the future, adaptive behavior may be supported so that the fit is affective, as well as cognitive and physiological. This would require monitoring emotions, which although not an easy task is certainly feasible. Similarly, adaptive behavior must be supported in social settings too.

The technological feasibility of supporting adaptive behavior in human-computer interaction is increasing dramatically, and so will the expectations of adaptive systems. As demonstrated above, in order to proactively recommend or even trigger adaptive behavior, the system must recognize the user's current position on dimensions of change and the current conditions. More and more sophisticated technologies (e.g., the Internet of Things) will be available to detect information about the user, the task and the setting. For instance, a user engaged in collaborating with remote others could be notified of the susceptibility of miscommunication at the receiver's end on the basis of sensors detecting noise in the communication. The system can then adapt the conversation appropriately.

More sensors transmit more information to users about the environmental conditions as well as the bodily conditions; most probably, people will feel pressurized to adapt accordingly. The new information technologies and infrastructures need, but also enable, more adaptation. The five-step procedure for designing systems that support adaptive behavior relies on the knowledge of the user, task and setting. The first step is to use the knowledge of the user's behavior when solving the task in order to determine the

dimensions of changing behavior, therefore, more knowledge and more sophisticated computing and data analytics will facilitate better support for adaptive behavior.

## References

- Annett, J.: Feedback and human behaviour: The effects of knowledge of results, incentives and reinforcement on learning and performance (1969)
- Billsus, D., Brunk, C.A., Evans, C., Gladish, B., Pazzani, M.: Adaptive interfaces for ubiquitous web access. *Commun. ACM* **45**(5), 34–38 (2002)
- Christoff, K., Keramiatian, K., Gordon, A.M., Smith, R., Mädlar, B.: Prefrontal organization of cognitive control according to levels of abstraction. *Brain Res.* **1286**, 94–105 (2009)
- Djamasbi, S., Siegel, M., Tullis, T.: Visual hierarchy and viewing behavior: an eye tracking study. In: Jacko, Julie A. (ed.) *HCI 2011. LNCS*, vol. 6761, pp. 331–340. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-21602-2\\_36](https://doi.org/10.1007/978-3-642-21602-2_36)
- Feigh, K.M., Dorneich, M.C., Hayes, C.C.: Toward a characterization of adaptive systems a framework for researchers and system designers. *Hum. Factors J. Hum. Factors Ergon. Soc.* **54**(6), 1008–1024 (2012)
- Hammond, K.R., McClelland, G.H., Mumpower, J.: *Human Judgment and Decision Making: Theories, Methods, and Procedures*. Praeger Publishers, New York (1980)
- Ilgel, D.R., Fisher, C.D., Taylor, M.S.: Consequences of individual feedback on behavior in organizations. *J. Appl. Psychol.* **64**(4), 349 (1979)
- Kennedy, M., Te'eni, D., Treleavan, J.: Impacts of decision task, data and display on strategies for extracting information. *Int. J. Hum. Comput. Stud.* **48**, 159–180 (1998)
- Louis, M.R., Sutton, R.I.: Switching cognitive gears: from habits of mind to active thinking. *Hum. Relat.* **44**(1), 55–76 (1991)
- Rasmussen, J.: *Information processing and human-machine interaction. An approach to cognitive engineering* (1986)
- Rittgen, P.: Negotiating models. In: *Proceedings of the 19th International Conference on Advanced Information Systems Engineering*, pp. 561–573. Springer, June 2007
- Ronen, H., Te'eni, D.: The impact of HCI design on health behavior: the case for visual, interactive, personalized (VIP) feedback. In: *ICIS 2013, Milan, 16–18 December, 2013* (2013)
- Sengupta, K., Te'eni, D.: Cognitive feedback in GDSS: improving control and convergence. *MIS Q.*, 87–113 (1993)
- Srinivasan, A., Te'eni, D.: Modeling as constrained problem solving: An empirical study of the data modeling process. *Manage. Sci.* **41**(3), 419–434 (1995)
- Sterman, J.D.: Modeling managerial behavior: misperceptions of feedback in a dynamic decision making experiment. *Manage. Sci.* **35**(3), 321–339 (1989)
- Sun, H.: Understanding user revisions when using information system features: adaptive system use and triggers. *MIS Q.* **36**(2), 453–478 (2012)
- Te'eni, D.: Analysis and design of process feedback in information systems: Old and new wine in new bottles. *Account. Manag. Inform. Technol.* **2**(1), 1–18 (1992)
- Te'eni, D.: Designs that fit: an overview of fit conceptualizations in HCI. In: Zhang, P., Galletta, D. (eds.) *Human-Computer Interaction and Management Information Systems: Foundations*, pp. 205–221. M.E. Sharpe, Armonk (2006)
- Te'eni, D., Sani-Kuperberg, Z.: Levels of abstraction in designs of human–computer interaction: the case of e-mail. *Comput. Hum. Behav.* **21**(5), 817–830 (2005)

- Timpf, S., Volta, G.S., Pollock, D.W., Egenhofer, M.J.: A conceptual model of wayfinding using multiple levels of abstraction. In: Frank, A.U., Campari, I., Formentini, U. (eds.) GIS 1992. LNCS, vol. 639, pp. 348–367. Springer, Heidelberg (1992). doi:[10.1007/3-540-55966-3\\_21](https://doi.org/10.1007/3-540-55966-3_21)
- Vallacher, R.R., Wegner, D.M.: What do people think they're doing? Action identification and human behavior. *Psychol. Rev.* **94**(1), 3 (1987)
- Vessey, I., Galletta, D.: Cognitive fit: an empirical study of information acquisition. *Inform. Syst. Res.* **2**(1), 63–84 (1991)



Group Decision and Negotiation. A Socio-Technical  
Perspective

17th International Conference, GDN 2017, Stuttgart,  
Germany, August 14-18, 2017, Proceedings

Schoop, M.; Kilgour, D.M. (Eds.)

2017, XII, 229 p. 41 illus., Softcover

ISBN: 978-3-319-63545-3