

An interaction model: from a user model to an environment model

Dr. Hokyoung Ryu¹
Prof. Andrew F. Monk²

¹Institute of Information and Mathematical Sciences
Massey University
Auckland, New Zealand
Email: h.ryu@massey.ac.nz

²Department of Psychology
University of York
York, England
Email: a.monk@yourk.ac.uk

Abstract

This paper presents an approach to modelling man-machine dialog, and the way the users tasks, interaction and feedback affect each other. The model is meant to support the concrete design of a user interface, based on descriptions of user's environment, goals and tasks. Throughout this paper, we assume that Human-computer interaction (HCI) can reasonably be understood as a continuous process of cyclic interaction between the user and the environment. The action the user takes leads to changes to the system or the environment. These are evaluated by the user, and then this evaluation results in changes to goals, and then the user takes another action based on the changes to goals. In order to effectively describe the continuous process of cyclic interaction, a dialog notation that a user interface designer could reason about the interactivity is needed. We claim that a cyclic notation is able to account for the intimate connection between goal, action and the environment, allowing a user interface designer to make explicit what a process achieves, as well as what triggers that process. It is thus possible for designers to build interactive versions of the designs so as to assess the assumptions made or being made regarding the interaction between the user and the system.

Keywords

Cyclic interaction, interaction model, user model, environment model, mode problem

INTRODUCTION

The practical user interface design process has often been dealt with at several levels. For example, Rasmussen (1986) and Vicente (1999) have concentrated on the job or task level, or the work environment of human-computer interaction. In contrast, Card, Moran, and Newell's study (1983) focuses on system users and their activities. Consequently, the general design process is usually considered as a phased structure that distinguishes logically separate activities. Different stages may employ different methods or representations for each phase. Monk (1998b) suggests a general design process as shown in Figure 1.

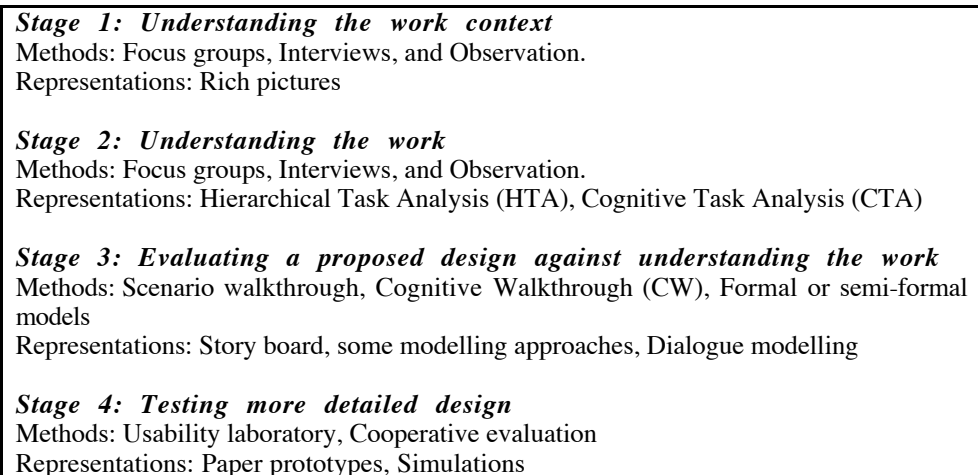


Figure. 1. A general design process – adapted from Monk (1998b).

The process starts with by obtaining an understanding of the general context for the work environment. In this stage, a description of the work environment relevant to interaction design is obtained for the intended user population. Interviews and focus groups provide ways of gathering data on users' preferences and specific work constraints and so on, whereas the designer can observe how the user employs the technology in their own work environment. A rich picture allows the designer to represent the work context in a more concrete style with some degree of abstraction. It also provides a flexible technique for systematic reviewing of the work environment and its control structure.

In the second stage, an understanding of the work environment results in a description of current major tasks or tasks to be redesigned. These will be used to design tasks to be evaluated in the third stage. In contrast with the first stage, several task analysis methods and use-case analyses can be used to identify critical tasks in the work environment. For instance, hierarchical task analysis (HTA) is one of the most well known forms of task analysis. It generally uses a graphical representation of a decomposition of a task into subtasks and operations, or actions. Whilst HTA is concerned with presenting a description of the steps that are required in order to achieve a task, cognitive task analysis (CTA) captures some representation of the knowledge that the user has or that they need to have in order to achieve a task.

Based on the first two stages, conceptual design alternatives of the major tasks are generated. In the third stage, these designs will be evaluated using walkthrough approaches, and some formal or semi-formal models.

Finally, in the last stage, the conceptual designs arising from the third stage are evaluated and modified through iterative evaluation techniques such as formal usability testing, in which representative users attempt to perform representative tasks with minimal training and intervention. This stage and the third stage are conducted in iterative cycles until all major tasks in the interface are covered.

This paper will give attention to the third stage, in particular, a conceptual model for evaluating a proposed design against understanding of the work. Early human-computer interaction (HCI) models have been successful for the third stage. For example, GOMS (Card et al., 1983) predicts well the goal-directed behaviour of users where there are predefined goal structures. TAG (Payne and Green, 1986) can be used, if the designer is able to understand the task features that must be in the mind of the user. Cognitive Walkthrough (CW: Polson et al., 1992) can evaluate hidden issues behind a design. For example, system effects that are insufficient to signal the completion of current goals. However, the assumptions made by the early HCI models are neither explicit enough nor usable enough for user interface designers with no background in HCI. Although we agree with the general findings from the HCI models if available, a main emphasis in this paper is given to assess interactions by conceptual modelling approaches.

Cyclic interaction theory

Most user-centred design approaches have concentrated on how the task set can be realized in the physical domain in which the user is engaged. To this end, most conceptual models of the user are concerned with how user's goals are translated into actions on the system (*goal-action paths*). Several dialogue models (e.g., action-effect rules: Monk, 1990, Harel, 1988) are concerned with how user's actions effect the system (*action-effect paths*). Consider a common task with a document on Microsoft Word™, 'Save a document'. GOMS notation may envision saving work on a PC as shown in Figure 2. It describes how user's goal will be decomposed into subgoals, and then mapped onto each action on the system.

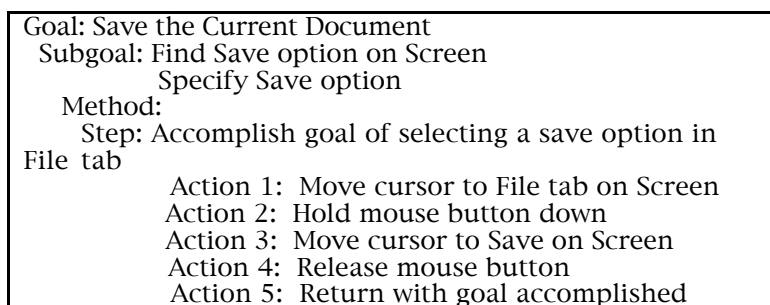


Figure 2. GOMS notation for a task 'Save the current document' on a PC – extended from GOMS notation (Card et al., 1983).

Yet, in GOMS, interactions are seen as very plan-driven with goal stacks and hierarchies. In this account, the interaction cycle between the user and the system is seen as starting with the user, even to the point where the effects of feedback are overlooked. This notation is not explicit enough for representing cyclic interaction. A complete account must consider both these two paths (i.e., *goal-action paths* and *action-effect paths*) and how changes in the system affect the user's goals (*effect-goal paths*), which is overlooked in the early HCI models. That is, the action someone takes leads to changes to the state of the world. These are evaluated with respect to, and in a manner

conditioned by, the user's current goal. This evaluation leads to the reformulation of goals and further action, leading to the new state of the environment, and so on.

Cyclic interaction theory (Monk, 1998a) may describe the saving task in a relatively comprehensive way for average designers. Here, the overall goal "save document" leads the user to take some action. As this is a GUI the user can scan the screen for something that might do this. The user recognizes the "File" menu tab. The user can click on it. We can describe that as taking the action "click on menu tab file". Clicking on the menu tab makes the display change, the menu drops down. This can be described as an effect. The visible effects of the user's action lead the user to change their goals. The goal "Save the Current Document" leads the user to generate the subgoal "Find Save option on Screen". The effect "File menu drops down" can lead the user to replace this subgoal with another subgoal "Specify Save option". The new goal set and new display state lead to a new action and the cycle continues. Figure 3 depicts a simplified mechanism of cyclic interaction. It characterizes these cyclic processes.

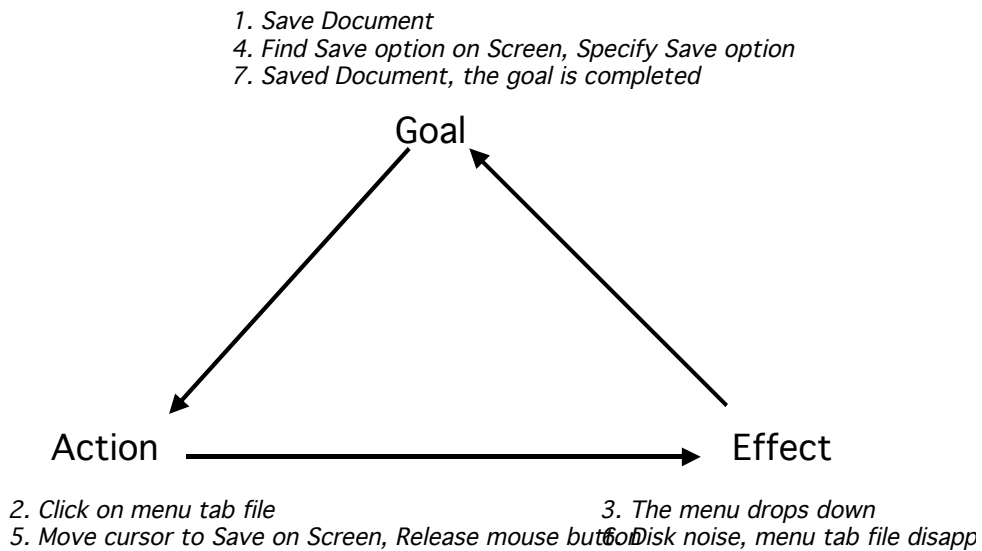


Figure 3. A cyclic interaction to save a document. the overall goal "1. Save Document" leads the user to take an action "2. Click on menu tab file". Clicking on the menu tab makes the display change "3. The menu drops down". The visible effects of the user's action lead the user to change their goals. The overall goal "1. Save Document" leads the user to generate the subgoal "4. Find Save option on Screen". The effect "3. The menu drops down" can lead the user to generate another subgoal "4. Specify Save option". The new goal set and new display state lead to a new action (5. Move cursor to Save on Screen, Release mouse button) and the cycle continues (6) until the overall goal is completed (7).

Although cyclic interaction theory can effectively describe interactions between the user and the environment, a critical factor, which will determine whether it achieves the uptake from average user interface designers, is the effort required by the designers to apply cyclic interaction theory to a practical design. In short, a useful notation and methods make cyclic interaction theory be more usable in designing interactions.

As a way of effectively handling the interaction modelling from the viewpoint of the designer, we introduce an interaction model called the 'Interaction Unit' to accommodate the perspectives above. Throughout this paper, we will refer to two features of the interaction model, the mechanism that the model employs and the benefit it provides in the design domain. However, this paper does not involve any evaluation work with HCI practitioners, which will be left to a future work.

INTERACTIONS BETWEEN THE USER AND THE ENVIRONMENT: CYCLIC OR ACYCLIC

Some HCI theorists (e.g., Norman, 1988, Howes, 1994, Howes and Payne, 1990, Kitajima and Polson, 1995, Monk, 1998a) have claimed that cyclic interaction models are able to depict how a user interacts with a system in terms of a continuous cycle with goals, actions, effects, and recognition. It can account for the intimate connection between goal, action and the environment. Goals in conjunction with some perception of the state of the environment lead to actions having some effects on the environment. These effects lead to changes in what is perceived and to new goals leading to new actions and so on.

Yet, some existing notations for cyclic interaction theory are less effective in representing the nature of cyclic interaction; for instance, either there is no notational scheme (e.g., Norman's seven stages model: Norman, 1988), or there are implicit notations concerning how system effects modify user's goals, e.g., D-TAG (Howes and Payne, 1990), Kitajima's theory (1995).

Combining the ideas addressed above, Ryu and Monk (2004) envisage cyclic interaction as shown in Figure 4, in order to model interactions explicitly with a more elaborated and conceptual manner. The general mechanism for the description of cyclic interaction in Figure 4 is as follows:

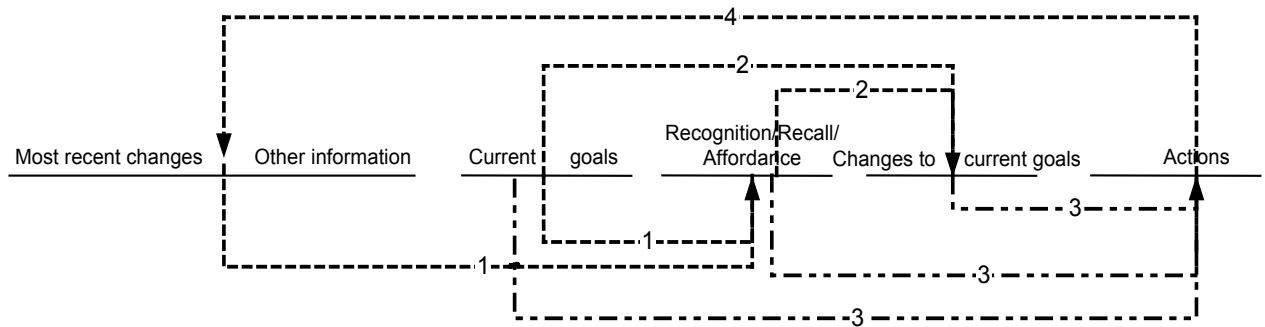


Figure 4 Capturing cyclic interaction in our interaction design framework. 1: Perception (Recognition/Recall/Affordance), 2: Goal reorganization process, 3: Goal-Action matching, 4: An action leads to the changes on the system or the environment.

Arrow 1 [Perception]: The user processes most recent changes, along with other unchanged information available about the system; this processing is conditioned by the current goals.

Arrow 2 [Goal reorganization process]: The processing results in changes to the current goals; this is also conditioned by the current goals.

Arrow 3 [Goal-Action matching]: The new goal stack obtained by adding the goal changes to the current goals leads to another action; the choice of action also depends on the processing in Recognition/Recall/ Affordance.

Arrow 4 [Action on the environment]: Some action by the user leads to some changes on the system, and the cycle continues.

One way of realising this idea is also portrayed in Figure 5 in a standard tabular format. It depicts how the cyclic interaction model can model the task “Save a file” which has been described with GOMS in Figure 2. How this works is explained in the next section.

	<i>Environment</i>		<i>User</i>			<i>Interaction</i>
	Most Recent Changes	Other Information	Current Goal	Recognition/Recall/Affordance	Change to Current Goal	Action
IU ₀			Saving a Doc.			
IU ₁	[START] Doc; MenuTab(File).	Mouse.	Saving a Doc.	Recall not saved Doc; Affordance Click MenuTab(File) --> Menu(File) Dropdown.	(+) Reveal command.	Click MenuTab(File).
IU ₂	Menu(File) Dropdown; when Click MenuTab(File).	Doc; Mouse; MenuTab(File).	Reveal command; Saving a Doc.	Recognize Menu(File-Save); Affordance Click MenuItem(Save) --> Doc Saved	(-) Reveal command.	Click MenuItem(Save).

AN INTERACTION MODEL: INTERACTION UNIT MODEL

Figure 4 and Figure 5 depict how cyclic interaction will be represented in conceptual interaction design. The main property of the model is a unit to describe a cycle of interaction. In this section, we will discuss how cyclic interaction is represented as a new unit – *Interaction Unit* (IU). The IU model describes interactions assumed by a designer using the new unit, making explicit how a user comes to take each action necessary to complete a task.

Interaction Unit (IU)

In the analysis and design of interaction, the three key elements are the user (or human), the environment (or the system or machine), and the interaction which takes place. The purpose of the model proposed here is to relate the three key components to interaction design with a range of theoretical modelling and analytic techniques. We believe this will be accomplished by the detailed description of both what triggers the interaction and what the interaction achieves.

Each *interaction unit* (IU), which encompasses both the description of goals and the inspectable states of the system, accounts for how the user will take each action. See Figure 5. The task ‘Save a Doc’ on Microsoft Word™ is achieved by two actions: ‘Click *MenuTab(File)*’ and ‘Click *MenuItem(Save)*’. Consequently, there are three interaction units to describe the task, including the two actions (IU₁ and IU₂) and the last interaction (IU₃) to check whether the task has been completed or not.

The development of an IU owes much to the works of Monk (1998a, 1999), who has claimed that cyclic interaction can be described in terms of both states and transitions. In his state transition framework for representing cyclic interaction, goals and effects are described as ‘states’ at the user and the environment respectively; action and recognition are described as ‘transitions’ (see Figure 6). Each transition describes interactions, beginning from a transition to trigger subsequent interactions and changing the state variables to define each interaction.

Another crucial philosophy behind the development of the state transition framework is that a cycle of activity starts with the state of the world and recent system responses with the user acting on the world in response to the current state. Consequently, the environment is described before the other descriptions are represented.

These two principles of the State-Transition Scenario (STS) are still the basis for an informal understanding of the interaction unit. In short, the concept of *interaction unit* (IU) is similar to the descriptive unit of the state transition framework, but provides a comprehensive account of the interaction. Consider Figure 5 and 6 that model the same task ‘Saving a file’. The first thing to notice is that the IU model is much more compact. This is because STS describes the two state-transitions, i.e., *action* and *recognition*. We argue that action is the more obvious focus for design than recognition, and recognition has to be considered as one of the cues for decision-making leading to the next user action.

The description using interaction units begins from the details of environments. See Figure 5. The first two columns

State of the environment	Physical states of the organism	Perceived states of the environment	Required states of the environment
STARTING STATE File not saved Menu tabs including ‘File’ visible	Oriented towards document	Document not saved	Document saved
1. ACTION (look at menus) File not saved Menu tabs including ‘File’ visible	Oriented towards menus	Document not saved	Document saved
File not saved Menu tabs including ‘File’ visible	Oriented towards menus	2. RECOGNITION Document not saved Menu tabs including ‘File’	Document saved Command visible
3. ACTION (select File) File not saved File menu including ‘Save’ visible	Oriented towards menus	Document not saved Menu tabs including ‘File’	Document saved Command visible
File not saved File menu including ‘Save’ visible	Oriented towards menus	4. RECOGNITION (note change) Document not saved File menu including ‘Save’	Document saved
5. ACTION (select Save) File not saved Menu tabs including ‘File’ visible	Oriented towards menus	Document not saved File menu including ‘Save’	Document saved
File saved Menu tabs including ‘File’ visible	Oriented towards menus	6. RECOGNITION (note change) Document saved Menu tabs	

Figure 6. State-transition scenario (STS) for ‘saving a file’ example. The changed states are given in bold – adopted from Monk (1999).

in IU_1 (IU_0 describes the initial condition, indicating the overall goal of the task) specify the inspectable part of the system state relevant to the action specified in the last column which is to “Click *MenuTab(File)*”. This can be thought of as a model of how the environment prompts a user to take some action. Inserted into this system model is a user component that specifies how the environment relates to the user’s goals.

Each interaction unit can thus be presented as a natural way of adding a user component to each part of a system model as represented in the program code. It allows the designer to develop interactive versions of their design so as to assess the assumptions made or being made by the designer about the interaction between the user and the system. In addition, it is likely that each unit may be appropriate to a formative analysis of the user interface.

Modelling interactions with Interaction units

We have discussed how the interaction unit can be used to represent interactions, and now it is time to turn our attention to how the designer can use interaction units in a practical design process.

The environment part of an IU is broken down into two parts: *Most Recent Changes* and *Other Information* (see Figure 5). These are changes as a result of the last action taken by the user or previous system states that can describe the environment (*Most recent changes*). These can also include a description of the environment given in the course of interaction (*Other information*). Interaction units are arranged in sequences to describe a scenario of use. Thus the first column ‘*Most Recent Changes*’ generally specifies the changes that resulted from the last action. IU_1 is the first IU in the scenario in Figure 5. As there is no most recent change the designer must specify what elements of the standing system state the user will use to prompt the appropriate action. For example, the most recent change ‘*Doc; MenuTab(File)*’ in IU_1 includes the initial information prompting the user; yet the most recent change ‘*MenuTab(File) Dropdown*’ in IU_2 is generated by the last action ‘Click *MenuTab(File)*’. While we assume that most IU will be triggered by a change in the environment there are occasions where unchanged inspectable effects influence user behaviour. These are listed in the column ‘*Other Information*’. These may be previous effects that condition the current IU. This concept refers to restricting user interactions that can take place at each interaction unit. This is a key aspect of understanding what the users will be doing when carrying out their tasks. For instance, ‘*Mouse*’ defines an input device or interaction style with which the user has to interact.

Viewing Figure 5 as a whole one can run down the last column (*Interaction/Action*) to see actions that form the scenario. The first two columns (*Environment*) describe the primary responses of the system to these actions. The other columns (*User*) flesh out the implicit assumptions of the designer about how the user comes to take their part in this interaction in terms of goal changes and processing the display.

The first column in the user part of the IU is the current goal stack (*Current Goal*). This changes during the scenario as subgoals are added or eliminated. The goal changes that are assumed to occur in the IU are specified along with the cognitive processes (*Recognition, Recall, Affordance*) resulting in these changes. The different roles of the cognitive processes will be described later.

Changes to goals are prefixed with a plus (+) sign that means newly generated goals or a minus (–) sign that indicates elimination of goals. Take IU_2 in Figure 5, the initial goal stack is ‘Reveal command’ and ‘Saving a Doc’. Recognition of the new display causes change to these goals, eliminating ‘Reveal command’. The new goal stack is thus ‘Saving a Doc’. An action following from this and the affordance presented by *MenuItem(Save)* on the keypad is to “Click *MenuItem(Save)*”.

How goals would be modified entirely depends on the user’s cognitive process. Consequently, to create a complete interaction model for average designers, it is necessary to describe these processes in a simple way for practical design decisions. IUs consider three cognitive attributes regarding how a user can proceed their interaction generating new goals or some actions taken. In other words, the IU notation focuses on the different roles of the three cognitive attributes: *Affordance, Recall, and Recognition*.

Objects in the system have *affordance* (Gibson, 1979, Norman, 1999, Djajadiningrat et al., 2002), which directly guides users’ expectations of what actions can be taken on the object and what effects these actions will have. In this paper, affordance is considered as recognition of an object that can be used to take some action and the effects that will result. For instance, in IU_1 of Figure 5, the recognition of ‘*MenuTab(File)*’ on Microsoft Word™ allows a user to push it. Also, ‘*Affordance Click MenuTab(File)*’ allows the user to expect that a desired effect will be accomplished, i.e., ‘*Menu(File) Dropdown*’.

In contrast, *Recognition* refers to the understanding of the meaning of the object. From the viewpoint of designers, it is necessary to distinguish affordance from recognition in that the affordance of an object strongly suggests the next action rather than recognition. *Recall* is considered as remembrance of an event occurred in the past. This has been less considered in the previous work such as display-based interaction models (e.g., D-TAG: Howes and Payne, 1990). Consequently there is no account of the interaction history used as a resource (Wright et al., 2000) which may characterize the selection of an action. These efforts to distinguish the three cognitive attributes emphasize how the designer’s expectation of the user’s behaviour can be encoded in objects and how the user can translate or notice the designer’s idea. These psychological considerations in the IU model do not require the average HCI designer to describe how this process can be achieved in greater detail as has been suggested for most user models. Rather, the designer describes what cognitive process of the three cognitive processes may be involved in performing interactions. These simplified cognitive processes detect differences between the current goals, the overall goals and the perceived state of the world, and thus trigger changes to the current goals. For example, the environment in IU_3 has ‘*Disk Noise; Menu(File) Disappear*’. When users recognize the effect and evaluate this with the current goal

stack, they can see that the overall goal has been achieved. In effect, the designer can simulate how the user will react or think in performing tasks using their proposed design. By asking the designer to conceptually describe just one way the user may complete the task, rather than a model that can cope with all contingencies, we further simplify the task. Figure 7 depicts general steps for describing interactions with interaction units.

In this section, we have seen how the IU model can describe cyclic interaction. The next section will discuss how the IU model can highlight some interaction problems in developing a user interface.

General Instructions in Describing IUs

1. Define the overall goal, e.g., 'Saving a Doc' in IU₀.
2. The overall goal is put in the *Current Goal* column in the first IU. Columns are to be read from left to right, in other words,
 - 2.1. The *Environment* along with the *Current Goal* leads to recognition. e.g., Both 'Doc' and 'MenuTab(File)' with a goal 'Saving a Doc' lead to 'Affordance Click MenuTab(File) --> Menu(File) Dropdown.' in IU₁.
 - 2.2. The *Environment* along with the *Current Goal* and *Recognition/Recall/Affordance* leads to the *Change to Current Goal*. e.g., 'MenuTab(File)' with a goal 'Saving a Doc' and 'Affordance Click MenuTab(File) --> Menu(File) Dropdown.' lead to the changes to current goal as 'Reveal command' in IU₁.
 - 2.3. The *Environment* along with the *Current Goal*, *Recognition/Recall/Affordance*, and the *Change to Current Goal* leads to the *Action*. e.g., 'MenuTab(File)' with a goal 'Saving a Doc' and 'Affordance Click MenuTab(File) --> Menu(File) Dropdown', and a newly generated goal 'Reveal command' lead to an action 'Click MenuTab(File)' in IU₁.
3. Add '[END]' in *Action* for the last IU. e.g., '[END]' in IU₃.

Descriptions of Environment and Interaction columns

Step 1. Descriptions of IU₁

Step 1-1. Add '[START]' in the *Most Recent Change* column in the first IU. e.g., '[START]'. As there is no most recent change in IU₁ the designer may specify what element of the standing system state the user will use to prompt the appropriate action.

Step 1-2. The *Other Information* column is also used to specify elements of the system the user must recognize that are not included in most recent changes, e.g., *Mouse* as the relevant input device with which the user has to interact.

Step 1-3. Describe an action in the *Action* column.

Step 2. Descriptions of other IUs

Step 2-1. Include observable changes on the system by the last actions in the *Most Recent Change* column. e.g., 'Menu(File) Dropdown' in IU₂.

Step 2-2. Include unchanged information on the system by the last actions in the *Other Information* column. e.g., 'Doc', 'MenuTab(File)' in IU₂.

Step 2-3. Take into account user-initiated actions in each IU. Each IU can have only one action or set of actions that can be considered as an action. The action is some combination of interactions with the system that lead to observable changes in the system.

Step 2-4. Return to *Step 2-1* until no more action can be taken in the *Action* column.

Descriptions of User columns

Step 3. Clarify current goals in the *Current Goal* column in each IU. e.g., in IU₁, current goal is the same with the overall goal. In IU₂, a goal 'Reveal command' that is generated in IU₁ is put in the *Current Goal* column in IU₂.

Step 4. Define the possible recognition/recall/affordance in the *R/R/A* column in each IU.

Step 5. Specify the process of goal construction-elimination in the *Change to Current Goal* column. e.g., the goal 'Reveal command' is popped out in IU₂.

Step 6. Return to *Step 3* until overall goal is completed.

Figure 7. General procedures to present an interaction scenario using the IU model – refer to Figure 5.

AN ILLUSTRATION: PREDICTING MODE PROBLEMS WITH INTERACTION UNIT

Consider for example a computer system where the effect of the actions taken depends on what mode it is in. The classic example is the text editor where in one mode keystrokes are interpreted as commands and in another as inserted text. Unless mode is clearly signalled people will make mode errors.

Modes give rise to two problems. The first is that a system may not immediately provide a user with the affordance to do what they want to do. This kind of mode problem is commonly observed when users interact with complex

systems. Ryu and Monk (2004) claimed that this kind of mode presence could be detected by examining the Most Recent Change columns in all IUs looking for different effects arising from the same action.

This case study is concerned with a second problem with modes, that is, that the user might not be aware that they are in the wrong mode in the first place. To take an everyday example, consider a remote control that can be used to work a TV and a Video player. The same buttons are used to control each appliance, which appliance responds to these actions depends on the mode the remote is in. This is changed by pressing the “TV” or the “VCR” on the remote control. In this case, when there is no clear mode signal provided, if someone wants to play a videotape they must remember the “VCR” button was pressed last. If someone else has used the remote control since you did, even this is impossible, resulting in much confusion. This is represented in IU₂ of Figure 8. That is, in order to figure out the current mode the user have to remember which button was pressed last. Such notations with recall, “Recall Pushed *Button(VCR)*”, will allow the previous action to change the perception of the environment and the current goal set at a given IU.

In effect, one should then examine the Recognition/Recall/Affordance columns in the IUs immediately preceding the IUs thus identified to check that a mode change is made. In this way, having written a conceptual IU modelling for some parts of a system the designer is asked to examine for IUs where the same action has a different effect indicating that the system has modes or the mode change has to be recalled.

	<i>Environment</i>		<i>User</i>			<i>Interaction</i>
	Most Recent Changes	Other Information	Current Goal	R/R/A	Change to Current Goal	Action
IU ₀			Play videotape.			
IU ₁	[START]	<i>Remote control.</i>	Play videotape.	Recognise <i>Button(VCR)</i> ; Affordance Push <i>Button(VCR)</i> --> <i>VCR Activate.</i>	(+) Change mode into VCR.	Push <i>Button(VCR)</i> .
IU ₂	<i>Nothing; when Push Button(VCR)</i>	<i>Remote control.</i>	Change mode into VCR; Play videotape.	Recall Pushed <i>Button(VCR)</i> ; Affordance Push <i>Button(Play)</i> --> <i>Videotape Play.</i>	(-) Change mode into VCR; (+) Select <i>Button(Play)</i> .	Push <i>Button(Play)</i> .
IU ₃	<i>Tape Play; Tape Noise; when Push Button(Play).</i>	<i>Remote control.</i>	Select <i>Button(Play)</i> ; Play videotape.	Recognise Played <i>Tape.</i>	(-) Select <i>Button(Play)</i> ; (-) Play videotape.	[END]

Figure 8. An IU model including recall. If someone wants to play a videotape, s/he has to recall button ‘VCR’ was pressed last.

CONCLUSIONS AND FUTURE WORK

The crux of the interface design is that a user interface designer is not able to find guidelines that cover every new technology. In this sense, this paper may shed light on the theoretical approach within the cyclic interaction design framework, which intends to support the designer in developing new interfaces. We believe that this provides an opportunity to reduce the development time of the new kind of interface rather than establishing all relevant design guidelines.

This paper addressed a valid problem, i.e., how to present an HCI model for design evaluation that can be useful to interface designers with little background in HCI. The work heavily relies on previous work of one of the authors (Monk, 1998a), but introduce the concept of an interaction unit. In detail, we have presented a model for the description of interaction, which is expected to be suited to a practical design process. Having introduced the interaction model, the *IU model*, we have identified differences between it and other HCI models from various stances. This has highlighted the advantages and limitations of the IU model in designing an interface. We have also seen that there are two bodies of work behind the interaction unit model. Firstly, the theoretical background of the IU model owes much to cyclic interaction theory. Second, the way of describing interactions is closely related to the principles of STS. It can also be seen that the IU model has advanced STS in order to provide a better ease-of-use model for designers with a more elaborated method. As an example, we illustrated how mode problems can be caricatured in modelling interactions. Similar procedures can be used to check other aspects of interaction units, for example, that the system prompts the user to establish and eliminate subgoals appropriately or that affordances prompt the right actions given a particular goal (Ryu, 2003). In this way, a designer can make fully explicit the assumptions made about why the user should interact in a particular way. In this process the design is refined and made mode usable.

The main contribution of the paper is a tabular notation for describing the user’s goal, the interaction the user interface provides for reaching this goal, what the user may choose to do, the feedback the system gives and how this in turn effects the user’s goal (i.e., cyclic interaction). This notation is supposed to be easier to read and analyse than a previous approach, State-Transition-Scenario (STS). Although the notation does seem to be simpler, it is not clear

how much simpler it is, and how much that matters for the design process. For this issue, we are planning an evaluation work to identify pros and cons of the notation with actual designers and report on their experience, which is carrying out to some extent.

REFERENCES

- Card, S. K., Moran, T. P. and Newell, A. (1983) *The Psychology of Human-Computer Interaction*, Erlbaum, Hillsdale, NJ.
- Djajadiningrat, T., Overbeeke, K. and Wensveen, S. (2002) *But how, Donald, tell us how? on the creation of meaning in interaction design through feedforward and inherent feedback*, *DIS 2002*, 285-291.
- Gibson, J. J. (1979) *The ecological approach to visual perception*, Houghton-Mifflin, Boston, MA.
- Harel, D. (1988) *On visual formalisms*, *Communications of the ACM*, **31**, 514-530.
- Howes, A. (1994) *A model of the acquisition of menu knowledge by exploration*, *CHI'94:Proceedings human factors in computing systems*, 445-451.
- Howes, A. and Payne, S. J. (1990) *Display-based competence - towards user models for menu-driven interfaces*, *International Journal of Man-Machine Studies*, **33**, 637-655.
- Kitajima, M. and Polson, P. G. (1995) *A comprehension-based model of correct performance and errors in skilled, display-based, human-computer interaction*, *International Journal of Human-Computer Studies*, **43**, 65-99.
- Monk, A. (1990) *Action-Effect Rules - a Technique For Evaluating an Informal Specification Against Principles*, *Behaviour & Information Technology*, **9**, 147-155.
- Monk, A. (1998a) *Cyclic interaction: a unitary approach to intention, action and the environment*, *Cognition*, **68**, 95-110.
- Monk, A. (1998b) In *"Lightweight techniques to encourage innovative user interface design"* (Ed, Wood, L.) CRC Press, pp. 109-129.
- Monk, A. (1999) *Modelling cyclic interaction*, *Behaviour & Information Technology*, **18**, 127-139.
- Norman, D. A. (1988) *The Psychology of Everyday Things*, Basic Books.
- Norman, D. A. (1999) In *ACM Interactions*, pp. 38-42.
- Payne, S. J. and Green, T. R. G. (1986) *Task-action grammars: a model of the mental representation of task languages*, *Human-Computer Interaction*, **2**, 93-133.
- Polson, P. G., Lewis, C., Rieman, J. and Wharton, C. (1992) *Cognitive walkthroughs - a method for theory-based evaluation of user interfaces*, *International Journal of Man-Machine Studies*, **36**, 741-773.
- Rasmussen, J. (1986) *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*, North-Holland, New York.
- Ryu, H. (2003) *Modelling cyclic interaction: an account of goal-elimination process*, *CHI 2003*.
- Ryu, H. and Monk, A. (2004) *Will it be a capital letter: signalling case mode in mobile devices*, *Interacting with Computers*, Revised.
- Vicente, K. J. (1999) *Cognitive Work Analysis: Towards Safe, Productive and Healthy Computer-Based Work.*, Erlbaum, Mahwah, NJ.
- Wright, P. C., Fields, R. E. and Harrison, M. D. (2000) *Analyzing human-computer interaction as distributed cognition: the resources model*, *Human-Computer Interaction*, **15**, 1-41.

ACKNOWLEDGEMENT

Major funding for this work was provided by Daewoo Electronics during the Ph.D. course of the first author. We would also like to thank the anonymous reviewers of the paper for their through comments and helpful suggestions, we are specially grateful to the member of the HCI group in University of York for their sincere comments of this paper.

COPYRIGHT

The following copyright statement with appropriate authors' names must be included at the end of the paper

[Hokyong Ryu/ Andrew Monk] © 2004. The authors assign to OZCHI and educational and non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The authors also grant a non-exclusive licence to OZCHI to publish this document in full in the Conference Papers and Proceedings. Those documents may be published on the World Wide Web, CD-ROM, in printed form, and on mirror sites on the World Wide Web. Any other usage is prohibited without the express permission of the authors.