Knowledge-based system based on consistency and congruity for logical and physical interface component design

Ho kyoung Ryu
IE Dept. KAIST, 373-1
Kusong-dong, Yusong-gu
Taejon , Korea
+ 82 42 869 3159
hogg@cogsys.kaist.ac.kr

Wan Chul Yoon
IE Dept. KAIST,373-1
Kusong-dong, Yusong-gu
Taejon, Korea
+ 82 42 869 3119
wcyoon@sorak.kaist.ac.kr

ABSTRACT

Previous user interface analyses have performed manually for evaluating complexity and consistency on user interface. However, manual analysis and evaluation process is not cost-effective and time-efficient. Additionally, interface problems in real life are dealt with by users whose knowledge plays a key role in understanding and using interface. That's why we need a new analysis and design method that could explicitly take user's expectation or user task knowledge into account.

This study expands this viewpoint with an automatic evaluation process and a knowledge-based system for it. The results guarantee an efficient and objective analysis and evaluation on user interface.

The implemented system (Evaluation System for Task Interface Matching ;ESTIM) explicitly represents a lot of user knowledge, logical interface component of user interface and then matches the two using knowledge-based system. In addition to, Extended ESTIM will cover physical interface element. Eventually, ESTIM not only evaluates the logical characteristics that are defined in the interface such as operation images, procedural consistency, and match with user's expectation, but also identifies the physical attributes such as labels of command button, size of push button and layout. Through this ESTIM result, designer and usability engineers can efficiently assess user interface. In this paper, diverse formalism and ESTIM will be demonstrated with a practical telecommunication appliance, pager.

Keywords

Consistency, Congruity, User task knowledge , Task-Interface Matching, Evaluation System for Task-Interface Matching, Knowledge-Based System, Logical interface component formalism, Physical interface component formalism

INTRODUCTION

Our life is surrounded with intelligent products of more complex and diverse function. Moreover, user would use products in different way that designer expect. In such case, committing errors, users experience frustrations and then resign rights to use high-cost function. That is why product designers should try to reflect user's needs into products. Up to now, in the studies on analysis and design of user interface, researchers have mainly studied the logical consistency, complexity or physical elements of user interface. However, users feel discomfort when a product has different behaviors with user knowledge, so we need cognitive evaluation method based on user knowledge for getting an interface understood. Under this motivation, we have developed a knowledge-based design system and novel evaluative rules based on diverse user knowledge on interface.

In the following sections, we will explain and validate a framework and support system through an example of pager.

FORMALISMS FOR USER INTERFACE ANALYSIS AND EVALUATION

The drawbacks of previous methods for analysis and evaluation of interface is that they take a lot of time and do not produce the consensus of result by analysts. That's why

we need a system which supports the analysis of user interface independent of analysts' ability and formalisms which can define user interface.

Formalism on logical interface component

It can be said that user interface has two disjoint component: Logical Interface, Physical Interface[4]. Therefore, our team designed different formalism for the interface modeling and evaluation: Logical Interface Component formalism(LIC formalism), Physical Interface Component formalism(PIC formalism). The detail explanation of each formalism will be shown in the following section.

The first step for analysis and design process is the representation of logical interface components on system. In this research domain, multiple approaches to formal specification of interface have been suggested[13][14]. However, this formalisms have some problems to apply to various case. Therefore, we developed a new formalism for specifying the logic of interface centered on user's action. It is called OCD[14]. OCD is the method for drawing cognitive process and representing users' action in performing a task. Figure 1 illustrates the entities for expressing logical interface components in OCD. And, Figure 2 exemplifies an OCD representation of task "Set date" in a pager.

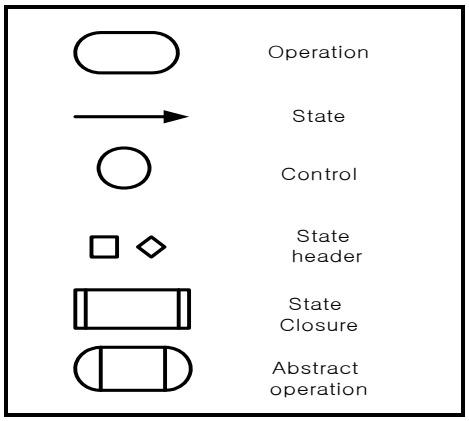


Figure1: LIC formalism -OCD

Going into details of Figure 2, it is shown user's action sequences for accomplishing a task "Set Date". Firstly, if button 'M' is pushed long, then menu icons as system response appear in the LCD of pager. This is state 'S1' of task "Set Date". At 'S1', user selects menu "Date", with scrolling by button 'M'. Also, at 'S1', user may want to go ready-state for recovering their error or other causes. For this behavior, user should wait for 3 seconds or push button 'M' and 'S' at the same time. In this way, LIC formalism is designed to be visual and intuitive by OCD.

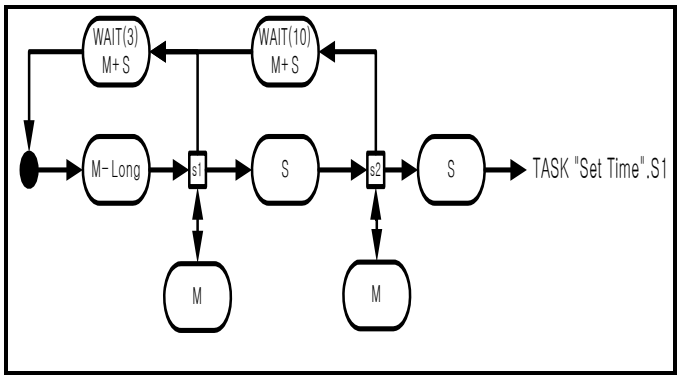


Figure 2: Diagrammatic LIC formalism of task "Set date" in pager.

Figure 2 do not illustrate system responses and function of actions, but they will be contained invisibly in each state. Especially, there is an editor to express them in ESTIM.
Also, diagrammatic LIC formalism as Figure 2 is transformed automatically into script LIC formalism in ESTIM. The following Figure 3 is a script formalism corresponding to Figure 2.

• Procedural Facts	• Function Facts	• Response Facts
(Sleep ; M-Long ; SH(S1))	SHOW(MENU)	SHOW(MENU) & BLINK(MENU)
(SH(S1) ; Wait(3)||M+S ; Sleep)	CANCEL	SLEEP
(SH(S1); M ; SH(S1))	CHANGE(MENU)	CHANGE(MENU)
(SH(S1); S; SH(S2))	NEXT	!BLINK(MENU) & SHOW(DATE) & BLINK(DATE)
(SH(S2); Wait(10)||M+S ; Sleep)	CANCEL	SLEEP
(SH(S2); M; SH(S2))	CHANGE(DATE)	CHANGE(DATE)
(SH(S2); S ;Task "Set Time". S1)	NEXT	!SHOW(DATE) & !BLINK(DATE) & SHOW(HR) & BLINK(HR)

Figure 3. Script LIC formalism of task "Set date" in pager

Formalism on physical interface component

All physical interface components can be represented by specified attributes. For example, command buttons have prior-defined attributes as the following: size, color, label, and label properties. These attributes are extracted from style guidelines[6][12]. ESTIM will contain the module for evaluation of physical interface components.

Formalism on users' knowledge

We need to acquire user's prior-knowledge, affinities, metaphors, analogies, and idiomatic term of operations for cognitive evaluation. Especially, our team divided task knowledge for users to use into four-level based on knowledge level: Means-end structures of tasks, Organization of operations, User's procedural knowledge, and Familiar patterns of controls [13][14][11].

Means-ends structures of tasks(MES)

The highest level structure of user task knowledge is means-ends structure that can be drawn in the form of goal tree [11].

Organization of operations.

Primitive operations are understood mainly in terms of two types of organization[14]: Semantic affinity of tasks, and Whole-part relationship. Affinity between subtasks and

operations makes user expect two aspects from an interface. First, the members with closer affinity should share more similar syntactical behavior. Second, similar tasks may well be included into a cluster or a mode. If an interface is not fit for supporting this user's expectation, user should memorize each task procedure in a cluster with separation. That's why this user task knowledge should be reflected in user interface design.

Additionally, users have knowledge of whole-part relationships among operations in a user interface. In general, users will memorize the consecution of operations as semantic units based on their knowledge with implication[14]. Accordingly, information or system responses which are distant from user's expectation will prevent user from making relationship of operations.

User's procedural knowledge

User knows some natural order of operations that are more or less generic to the task or got through interaction with different products or systems. There are three important procedural knowledge structures: Sequence, Branch and Loop knowledge structure. In contravention of these user expectation, user may feel uncomfortable with extant user interface[14].

Familiar patterns of controls

There is a few well-known control patterns as idioms in user interface. Toggle button is one of them. If the user interface do not allow these 'take for granted' knowledge of users , additional cognitive overload will be required[14].

Overview of analysis and evaluation of user interface

Analysis and evaluation on user interface can be performed from diverse viewpoints[3][5][9][10]. These viewpoints may produce inconsistent analyses or require a long time to complete an evaluation on interface. Therefore, Yoon proposed a time-efficient and cost-effective cognitive approach, which is called Task-Interface Matching(TIM), based on LIC formalism and UK formalism[13]. The following Figure 4 illustrates the framework of TIM.

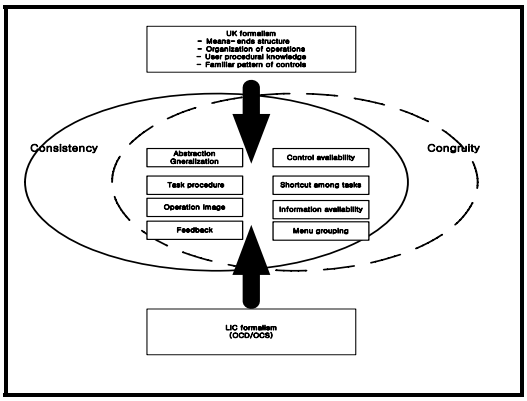
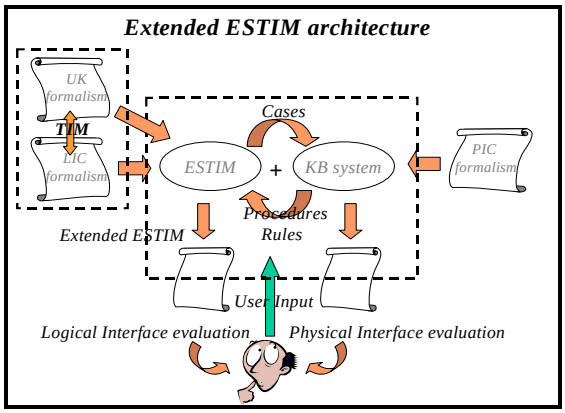


Figure 4:Framework of TIM using LIC and UK formalism.

Especially, PIC formalism which extended in this paper is used as input facts in knowledge-based system. The following figure illustrates an overview of extended ESTIM.



ESTIM: EVALUATION SYSTEM FOR TASK-INTERFACE MATCHING SYSTEM

Representation of LIC formalism in ESTIM

We developed the system for supporting the analysis framework illustrated in Figure 4, which is called ESTIM. In ESTIM, interface representations are depicted as Figure 5. In Figure 5, LIC formalism can be expressed by OCD format or form-filling editor.

Additionally, abstraction and generalization process of some operation set is represented in Figure 6. In the first place, user selects operations, states, and state headers building up an abstract operation or state closure. And then, the selected entities are assigned each name and function(s).

Also, each operation produces a change of system response and has peculiar function. In ESTIM, Figure 7 illustrates a form of expression on system response(s) and function(s).

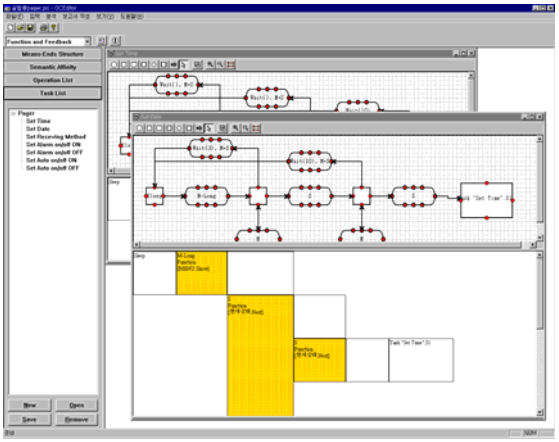


Figure 5:The window for representing LIC formalism in ESTIM

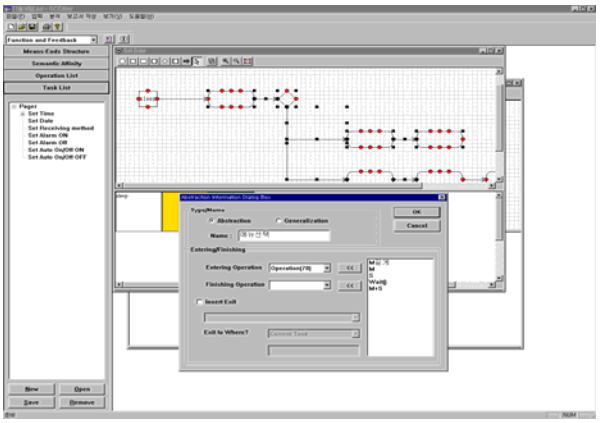


Figure 6: Abstraction and generalization process of logical interface components in ESTIM.

Representation of users' knowledge in ESTIM

In ESTIM, there are various forms of expression of user knowledge. First, representation of MES is given in Figure 8. For example, in USINE [5], Lecerof proposed that temporal relationship of tasks is classified into 7 types.

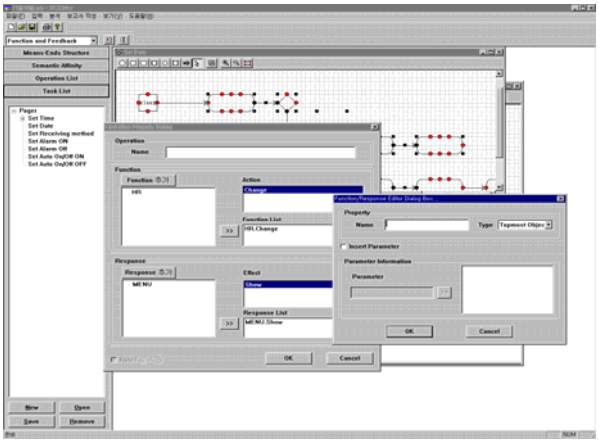


Figure 7: Expression of system response(s) and function(s) in ESTIM

In this paper, we proposed that temporal relationship of tasks falls into 5 types: unordered, parallel, sequence, alternative and optional types.

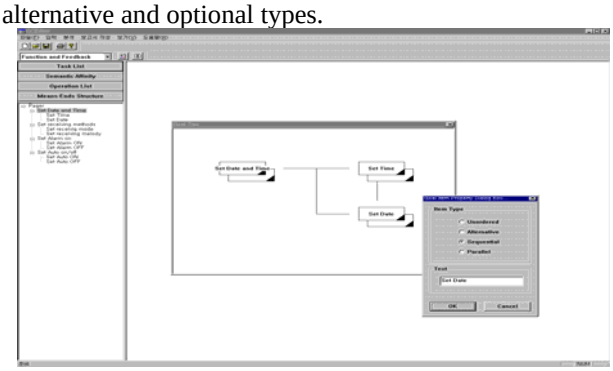


Figure 8: MES of task " Set Date and Time" in pager example.

Second, in ESTIM, it is implemented a modified clustering analysis for getting affinity knowledge among tasks with data in Figure 9[8]. These data are acquired by grading as 3-point scale(Low, Medium, High related) on each task couples. After getting data, through formulas we can automatically get all affinity indexes between tasks. In addition , this result is visualized by graphical type in Figure 10. In ESTIM, this semantic affinity knowledge of task will be utilized for evaluating congruity of menu structure and selection of tasks which should have similar syntactic behaviors.

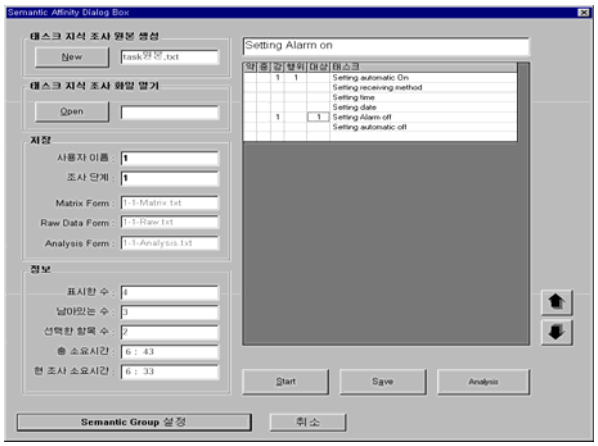


Figure 9: Comparison data among task in ESTIM

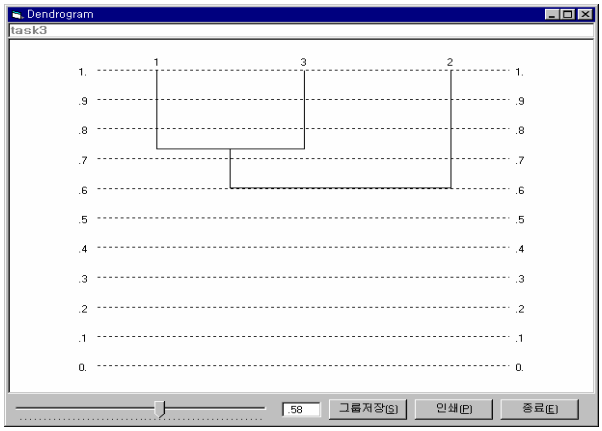


Figure 10: Graphical presentation of semantic affinty index in ESTIM.

Third, In ESTIM, we proposed function diagram for representing the user's procedural knowledge. This is based on function analysis which describes the events that must occur for user to achieve intended results. It doesn't describe physical actions that users will take or specify system actions[15].

Function diagram is made up of 'Function', 'Decision node', and 'Time flow'. For example, in pager, function diagram of task "Set ALARM ON" will be represented as the following Figure 11.

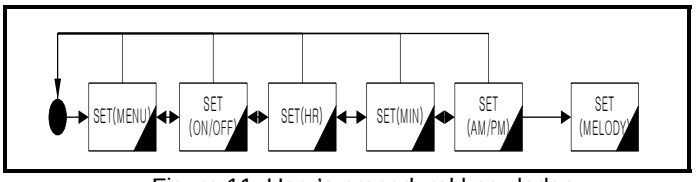


Figure 11: User's procedural knowledge of task "Set ALARM ON"

In ESTIM, there is a window for drawing function diagram as Figure 12.
Eventually, this UK formalism may serve as reference materials to enhance the understanding of human-system interaction, or they can be used directly to identify training needs and contents [15].

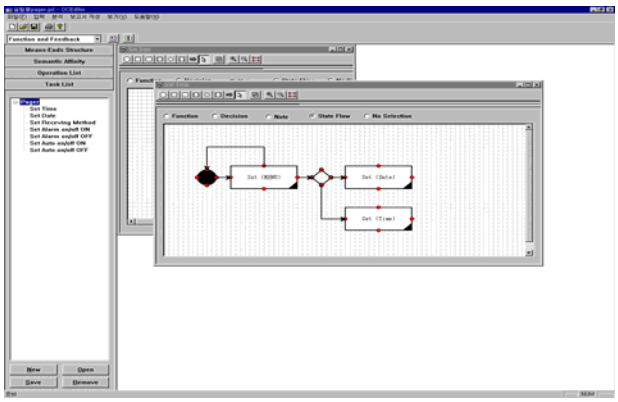


Figure 12: Function diagram for representing the user's procedural knowledge in ESTIM

TASK-INTERFACE MATCHING: CONSISTENCY AND CONGRUITY

Our main viewpoint of interface analysis and evaluation is consistency and congruity. Consistency is the most widely used measure for goodness of user interface. This is because consistency greatly reduces complexity of required knowledge, and can be measured with relative objectivity. In case of TAG, it tries to measure the consistency of procedure, but it does not consider the system response[2]. But users expect same response(s) for the same operation at a similar situation, consistency check of tasks should include system response(s). In addition, it is needed to consider the functional consistency of operations. Because operation has peculiar function(s) as means to achieve the goal, each operation should have same function(s) in task space. Otherwise, users may be confused by interpreting function in every action. Additionally, similar tasks need to have a similar procedure, because user will understand the interface as simplified procedures.

The following analysis process is performed based on LIC formalism and UK formalism.

Operation image

Operations are classified into two categories. One is a primitive operation for reaching to particular goal in performing tasks. The other is a navigational operation, so-called control, for reaching to particular state. Therefore, we have two types of operation image: Primitive operation image, and Navigational operation image. Moreover, primitive operation image is divided into two types from the orientation-viewpoint: Function-oriented, and State/response oriented. We have implemented analytic rules to evaluate a function-oriented operation image in ESTIM and LISP. Figure 13 and 14 illustrates a function-oriented operation image of a pager in ESTIM and LISP environment. Figure 15 shows a navigational operation image of a pager in LISP environment.
From Figure 13,14, we can identify inconsistency point that action "change" is performed by different operations, 'M' or 'S' in task space.

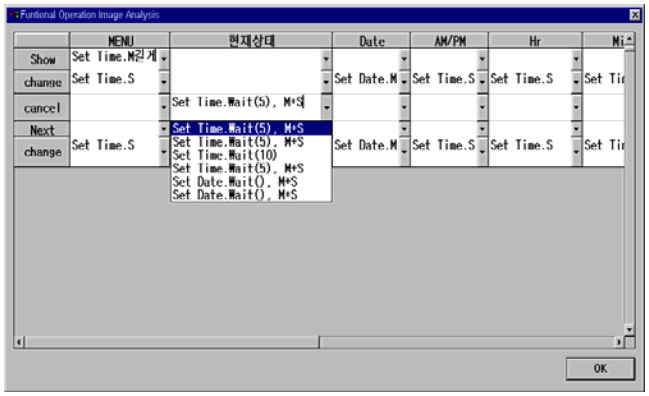


Figure 13: Example of function-oriented operation image analysis in ESTIM environment

---
"Set Date" Opeartion analysis

(M -> CHANGE(X)).
(M+S -> CANCEL(X)).
(WAIT(X) -> CANCEL(X)).
(M-Long -> SHOW(X)).
(S -> Set Time (X)OR NEXT using --> Inconsistency).

"Set Time" Opeartion analysis

(M+S -> CANCEL(X)).
(WAIT(X) -> CANCEL(X)).
(M-Long -> SHOW(X)).
(S -> NEXTOR CHANGE(X) using --> Inconsistency).
(M -> CHANGE(X)OR NEXT using --> Inconsistency).
(M -> CORRECT(X)OR CHANGE(X) using --> Inconsistency).

Figure 14: Example of function-oriented operation image analysis in LISP environment

Procedural consistency of similar tasks

Similarities between tasks make user expect the similar procedures or grammatical behaviors. Therefore, we need to evaluate whether similar tasks have the similar behaviors or procedures or not. In ESTIM, we use a modified cluster analysis for specifying the similar tasks, the result is

depicted in left tree-view of Figure 16. Specified tasks to be similar by UK formalism transform into script LIC formalisms. With those data, full comparison of tasks will be performed for finding inconsistent tasks

.

```
Abstract Operation: 'Set MIN'
Task      Entering   Finishing      Exiting              Exiting-State
AUTOON    NONE       NONE           (WAIT 10 OR M+S)     SLEEP
ALARM     NONE       NONE           (WAIT 10 OR M+S)     SLEEP
 =          =          =              M                    X1
TIME      NONE       NONE           (WAIT 5 OR M+S)      SLEEP

<<< Consistency Check in AO 'Set MIN' >>>
ENTERING :: CONSISTENCY
FINISHING :: CONSISTENCY
EXITING-STATE :: INCONSISTENCY
- Task 'TIME', 'ALARM', 'AUTOON'--> 'AO: Set MIN' 's EXITING-STATE is 'SLEEP'
- Task 'ALARM'--> 'AO: Set MIN' 's EXITING-STATE is 'X1'
Exiting INCONSISTENCY: Exiting operation's number is different in Task!
Exiting operation's parameter INCONSISTENCY: operation 'WAIT' 's parameter is different!
```

Figure 15 : Example of navigational operation image in LISP environment

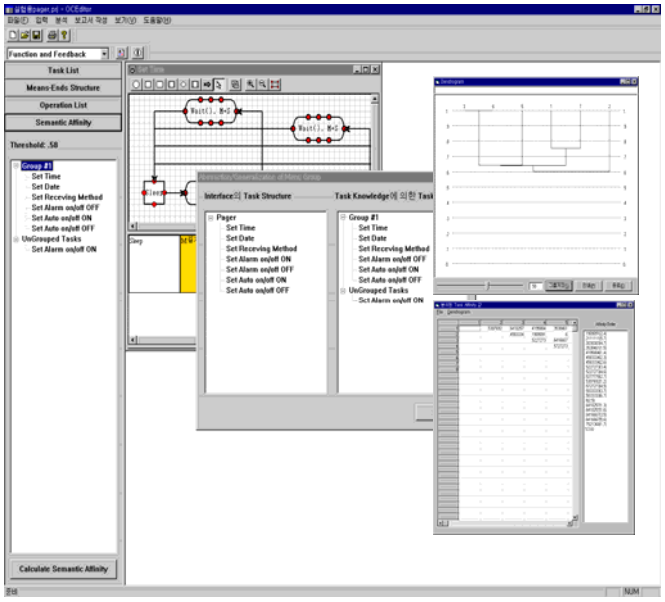


Figure 16: Specification of tasks to be similar by UK formalism in pager

User has some procedures or goal tree or other prior knowledge about tasks that are more or less generic. We called this 'user task knowledge'. This stage, in congruity analysis, shows that how well logical interface component(LIC formalism) is in harmony with user task knowledge(UK formalism).

Matching a logical interface procedure with users' procedural knowledge

The first step in congruity analysis is to check whether procedural knowledge is in harmony with the procedure of logical interface or not. For example, we came to know that there are procedural incongruities in pager as the following Figure 17

```
TASK "SET ALARM ON"
 <User's expectation>
(SLEEP -> (SET MENU (ALARMON)) -> ONOFF -> SET_TIME ->
SET_MIN -> SET_AMPM)
 <System behavior>
(SLEEP -> (SET MENU (ALARMON)) -> SET_TIME -> SET_MIN ->
SET_AMPM -> ONOFF)
 <User's expectation>        <System behavior>
-------------------------------------------------------------------
 (ONOFF-> SET_TIME)            (NO)
 (ONOFF-> SET_MIN)             (NO)
 (ONOFF-> SET_AMPM)            (NO)
```

Figure 17: Example of incongruent procedure with user's expectation of pager in LISP environment

In Figure 17, task "Set Alarm On" is composed of five abstract functions: Set(MENU(ALARM ON)), Set(ONOFF), Set(TIME), Set(MIN),and Set(AM/PM). We assume that in real product the sequence of these functions are in turn Set(MENU(ALARM ON)), Set(TIME), Set(MIN), Set(AM/PM), and Set(ONOFF). In such case, through depth-first search with LIC formalism and UK formalism, ESTIM highlights the differences in user's expectation and system's behavior.

Control availability

The mismatch between UK formalism and LIC formalism will cause error in performing tasks. Especially, when users expect a particular action in a specific state, however not implemented in a real system, this may invoke some problems. Therefore, availability of operations in the particular state is an important issue in the error-prevention of interface. For example, in Figure 18, users think that after "Selecting Menu: Alarm on ", "Selecting ON/OFF" can be done. But, in real product, "Selecting ON/OFF" can be performed at last time. Therefore, it is natural that users are prone to error in that point of control unavailability.

```
TASK "SET ALARM ON"
**************************
Control Availability Test
**************************
User expectation: ((SET MENU (ALARMON)) -> ONOFF)
System behavior: (((SET MENU (ALARMON)) -> SET_TIME -> SET_MIN
-> SET_AMPM ->   ONOFF))
Result: INCONSISTENCY

User expectation: (ONOFF -> (SET MENU (ALARMON)))
System behavior: ((ONOFF -> SLEEP -> (SET MENU (ALARMON))))
Result: INCONSISTENCY

User expectation: (ONOFF -> SET_TIME)
System behavior: ((ONOFF -> SLEEP -> (SET MENU (ALARMON)) ->
 SET_TIME))
Result: INCONSISTENCY
```

Figure 18: Control availability of pager in LISP environment

Shortcut among tasks

Users may have an idea of necessity for tasks to be linked.

In pager example, after user sets up receiving method as MELODY, users may usually want to decide MELODY TYPE. That is, user thinks that two tasks are closely connected by higher level goal. Therefore, we implemented an analytic rule so as to evaluate the differences with user's expectation in navigational availability.

Menu grouping

Nowadays the greater part of electronic or software system is menu-based than before. In these system, required behavior to complete a task is how to select menu items, and how to recognize where the item is located. Thus, one of sufficient conditions of good interface is a menu structure matching user's expectation. The transformation of UK formalism is illustrated in Figure 19. From this Figure 19, we can identify that users expect that task "Set ALARM on/off ON" is located in a different position, but it is included same position in an interface.

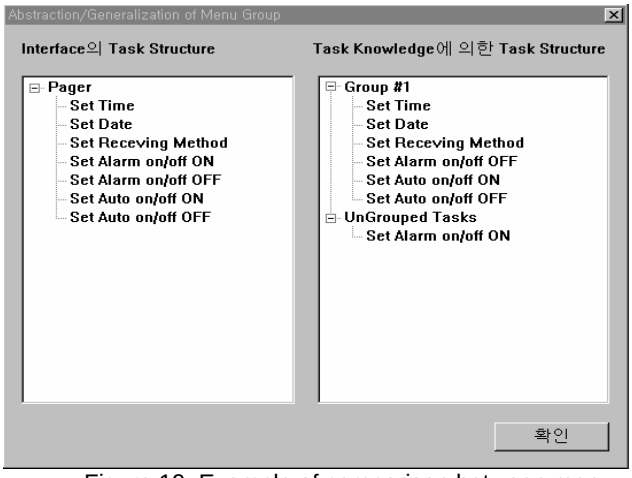


Figure 19: Example of comparison between menu structure of interface and one by user knowledge

Manual evaluation and automatic evaluation based on ESTIM

We found the facts that ESTIM proved to be more time-efficient and the number of highlighted problem found matched up to about 95% in comparison to manual analysis.

PIC FORMALISM AND KNOWLEDGE-BASED SYSTEM

PIC formalism of user interface contains the layout information which defines external and visual form of each component and the organization information which specifies relationship and grouping among components. Our team is building the layout rule and organization rule for evaluating of physical interface component respectively.

CONCLUSION AND FUTURE WORKS

We have progressed in development of novel method for user interface analysis and evaluation. Ultimately, the most important perspective on interface design is congruity with UK formalism as well as consistency. Therefore we should have the expression for UK formalism, LIC formalism, and PIC formalism. Also, we have implemented a knowledge-based system to support these cognitive evaluation process. We validated that the results through knowledge-based system are equivalent to manual analysis under TIM framework.

In future, through a novel acquisition and representation method about UK formalism and exquisiteness of inference engine on physical interface component, this study will be expanded to a total interface analysis tool. Additionally, ESTIM will be expanded to user interface management system.

REFERENCES

1. Byrne. M.D, Wood. S.D, Foley.J.D, Automating interface evaluation, In *Proceeding of ACM/CHI '94*, pp.232-237
2. Harrison. M and Thimbleby. H, Formal methods in Human-Computer Interaction, Cambridge, 1988
3. Kieras.D.E, Wood.S.D, Meyer D.A, Predictive engineering models based on EPIC architecture for Multimodal high-performance human-computer interaction task, *ACM Tran. On Computer-Human Interaction*, 4,9(Sep, 1997),pp230-275
4. Kim .W.C, Foley.J.D, Providing high-level control and expert assistance in the user interface presentation design, *INTERCHI '93*
5. Lecerof. A, Paternó. F, Automatic support for usability evaluation, *IEEE Tran. On software*, 24,10(Oct. 1998),pp.863-888
6. Löwgren. J, Lauren.U, Supporting the use of guideline and style guides in professional user interface design, *Interacting with computer*, Vol. 5, 1993, pp. 385-396
7. Löwgren. J, Nordqvist.T, Knowledge-based evaluation as design support for GUI, *CHI '92*
8. McDonald.J.E, Stone. J.D, Liebelt.L.S, Searching for items in menus: The effects of organization and type of target. In *proceedings of the 27th Annual Meeting of Human Factors Society*, pp834-837
9. Nielsen. J, Usability Engineering, American Press, 1993
10. Palanque.P, Paterno.F ,Bastide, R. Mezzanotte.M, Towards an integrated proposal for interactive systems design based on TLIM and ICO, *Design, Specification and verification of Interactive Systems '96*
11. Rasmussen.J, Pejterson. A.M, Goodstein.L.P, Cognitive system engineering, *Johns Wiely and sons*, 1994
12. Scapin D.L, Organization human factors knowledge for the evaluation and design of interface, *Int. J. of Human Computer Interaction* ,vol. 2,1990, pp.203-229
13. Wan C. Yoon, Jisoo Park, User interface design and evaluation based on task analysis, In *proceeding of ICPR '97* pp. 598-601
14. Wan C. Yoon, Jisoo Park, An interface model for evaluating Task-Interface congruity, In *proceeding of HCI international '97* , pp. 295-298
15. Williams.E , Rideout.T, Task analysis in the product design, Hewlett Packard