Volume 4, Issue 10, October 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Windows User Workspace: Quality Considerations and User Satisfactory Measurement

Bakpo, F. S.
Department of Computer Science
University of Nigeria, Nsukka, Nigeria

Ukekwe, E. C
Department of Computer Science
University of Nigeria, Nsukka, Nigeria

Abstrasct: The paper describes the notion of windows user workspace (WUW) as a state- of the-art representative of human computer interface (HCI) design. The aim is to provide understanding on important supportive tools and quality considerations that must be addressed by windows workgroup as well as measuring user's satisfaction on those facilities. Three general features frequently mentioned to address quality considerations (i.e., user modeling, adaptation and facilitation) are surveyed. Furthermore an empirical assessment is conducted on user satisfactory remarks and the open issues/problems with windows user workspace..

Keywords: Windows user support, human-computer interactions, quality considerations, factor analysis, survey, varimax and principal component analysis

I. INTRODUCTION

Windows (i.e., partitioning the display into a number of virtual screens) are extremely useful and powerful means for user communication with the computer. Windows user workspace was introduced under the auspices of the Windows operating systems to automate the low-level decision making of computers while leaving the users to interact at a higher level. It out weights Disk operating system (DOS) in that while DOS provides predominantly command - line interface, Windows OS provides in addition to command line, other user friendly interfaces such as menus, graphical user interfaces, etc. Windows provides a typical state -of- art representative of HCI design requirements. The challenge in Human Computer Interface (HCI) design is not only to design and build user interfaces that are transparent so that the highly functional and intelligent systems can be made useful without appearing complex or difficult to operate, but to help bridge the gaps between the user and the system during the operation of the system. The purpose of the system is to serve the user; hence the needs of the users should dominate the design of the interface, while the needs of the interface should dominate the design of the rest of the system [1]. Furthermore, user workspace must be seen as an integral run time element of the user interface. User interface is critical to the success of products in the market place, as well as the safety, usefulness, and pleasure of using computer-based systems. The intellectual foundations of user interface derived from a variety of fields: computer science, cognitive psychology, perceptual psychology, linguistics, artificial intelligence, and anthropology [2,3]. Several researchers have done commendable works in these fields. For an extensive review, the reader is referred to [1,2,3,4,6]. The topic of Windows user Workspace is extremely broad but we will restrict ourselves to the computer-based support of individual users during and throughout their interactions with the computer. We are interested in the features and facilities that are provided by Windows Workspace as well as the pertinent quality considerations, issues and problems.

II. HUMAN - COMPUTER INTERFACE DEFINED

We humans communicate by means of symbols to which we ascribe meanings. For example, we use many kinds of symbols: characters, numbers, diagrams, icons, spoken words, music, facial expressions and gestures. Computer science may be defined as the study of the automation of symbol processing [5,7]. This involves the design and construction of symbol processing machines (hardware) and of the instructions that control the machines (software).

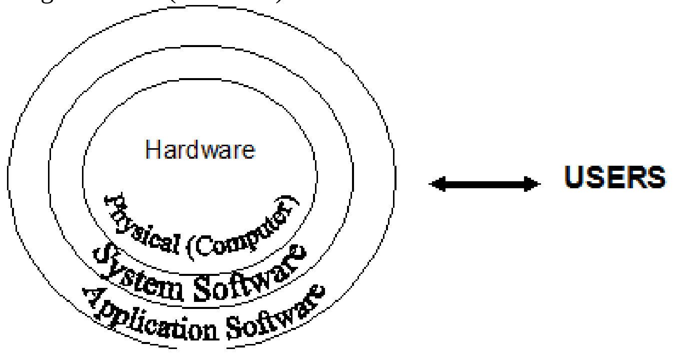


Fig.1 Relationship between the Computer and the User

As shown in figure 1, the user communicates with the computer via an interface at the outermost layer or application layer software known as human-computer interface. Human-Computer Interface (HCI) may be defined as the set of devices used to communicate machine concepts to a human user and vice versa [3,4,5]. It involves input and output devices such as keyboards, mice, video screens and the interaction techniques that use them such as how information is presented and requested, how the computer's actions are controlled and monitored, all forms of help, documentation, and training; the tools used to design, build, test and evaluate user interfaces, and the processes that developers follow when creating interfaces. HCI always accomplish two fundamental tasks: communicating information from the machine to the user, and communicating information from the user to the machine. All systems that communicate in some way with a user have a user interface. The body of science responsible for studying this is known as Human-computer Interaction (HCI). The following types of human-computer interfaces can be identified:

A. Graphical User Interface

A graphical user interface (GUI) acts as a friendly "front-end" making it easier for users to interact with the computer using graphical symbols on the computer screen called icons or metaphors [3,4,10]. These icons represent commands, which one would have typed at the command prompt in Disk Operating System (DOS) for example. The most pervasive element used in GUIs is the window. The video screen itself can be considered to be a window into the computer. The GUI paradigm actually allows the user to see into multiple areas within the computer. This greatly reduces the amount a user has to learn and remember. Figure 2 depicts a GUI.

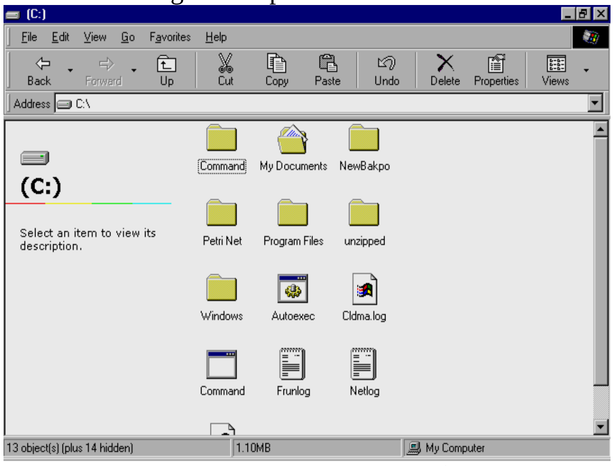


Fig. 2: A graphical user interface

B. Natural language interface

This refers to the ability to simply talk to a computer using words and have it fulfill a request. The goal of natural language processing therefore is to minimize the training required for users to use a system. The more naturally users can express their information needs in plain English, the fewer burdens upon them to learn that system. Such interface is measured by at least four criteria [9], namely conceptual, functional, syntactical and lexical. The conceptual domain describes the range of language covered by the system such as the information that is available in the database. The functional domain describes the limits of how questions can be asked of that system. The syntactical domain refers to the number of paraphrases of a single command that the system understands. Finally, the lexical domain refers to the total number of words that are contained in the system's lexicon.

C. Intelligent agents

This refers to software component (service program) that executes autonomously, communicates with other agents or the user, and monitors the state of its execution environment [7]. They are a new class of software that acts on behalf of the user to solve complex problems. The central notion underlying software agents is that of delegation. The owner or user of software agent delegates a task to the agent and the agent autonomously performs that task on behalf of the user.

D. Direct manipulation techniques

The basic idea behind the concept is to transform data into graphic representations that will help users understand the information they have received from the machine [11]. This concept takes the basics of GUI and pushes the limits of design to include such things as 3D imaging, filter-flow modeling, and dynamic query.

III. QUALITY CONSIDERATIONS FOR USER SUPPORTIVE INTERFACES

The objective of any user interface developer is to design and implement quality user interfaces [5]. Unfortunately, it is not always that easy to define what is meant by a "quality user interface". For the purpose of this paper, a quality user interface is defined as any user interface that is intuitive, easy to use, and allows the users to maximize their efficiency and effectiveness. The following qualities must be considered in the design of a good user supportive interface.

A. Intuitiveness

The most fundamental quality of any good user interface should be that it is intuitive. An intuitive user interface is one that is easy to learn. The use of icons, text labels, objects, radio buttons, etc can help to make a user interface intuitive. Some examples of intuitive elements are shown in figure 3.

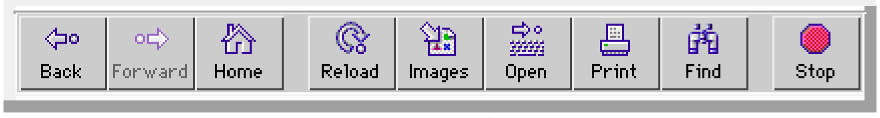


Fig.3: Examples of intuitive elements

Labels should be concise, cogent, and unambiguous. A good practice is to make labels conform to the terminology of the business that the application supports. This provides a good way to pack a lot of meaning into a very few words.

B. Adaptability

An adaptable user interface is defined as an interface that:

- Supports a number of different dialogue modes (e.g. menu-type and the command-language-type). More than two modes may also be provided;
- Allows the user to switch between dialogue modes at any time; i.e. even in the middle of a command;
- Makes the switch between dialogue modes smoothly and naturally;
- Makes it easy for the user to learn how to use the different dialogue modes.

Realizing the different needs of beginners and experts, many systems provide this quality, that is, the use of two distinct user interfaces: **a menu-driven** and **a command language driven**. Figure 4 shows a pull-down menu.

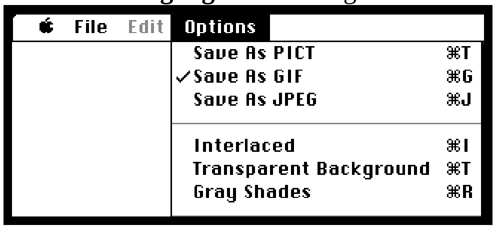


Fig.4: A Pull-down menu/Drop-down menu

Pull-down menus are menus that the user can “pull- down” from the menu bar that traverses the top of the screen. This type of quality provision is also loosely called **user-friendliness**, because the system supplies the user with available options in an application from which he makes his choices.

C. Consistency

The use of labels, icons, metadata, standard multimedia formats, web browsers, etc must always be consistent. They must always mean the same thing. Consistency must be maintained between applications and systems. User interface developers should always provide permanent objects as unchanging reference points around which the users can navigate. If they ever get lost or disoriented, they should be able to quickly find the permanent objects and from there get to where they need to be. Most of all Microsoft applications provide the menu bar and toolbar as examples of permanent objects (refer to figure 5).

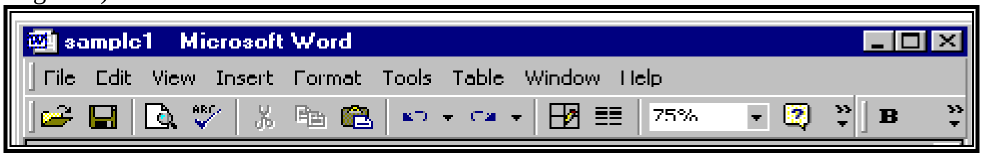


Fig. 5: Microsoft permanent objects

No matter what application the user is in, those objects will appear on the screen.

D. Simplicity

The best interface is one, which will be simple, apparent and effective to every user. A good gauge of simplicity is often the number of panels that must be displayed and the number of mouse clicks or keystrokes that are required to accomplish a particular task. The hackneyed programming principles called Keep It Simple Stupid (**KISS**) should be adhered to when designing user interfaces. The fewer things users have to see and do in order to get their work done, the happier and more effective they will be.

E. Visibility

Poorly designed systems often hide important functions, obscuring the reasoning behind the system’s actions. Users should be able to see what options they have and how to perform, even before they execute a task. The easiest way to make things visible and understandable is through the use of graphics and pictures.

F. Constraints handling

A good user supportive system should make the user feel as if there is only one possible thing to do with the system and that whatever he is doing is the correct action. Clear boundaries around the limits of the system will give the user a better sense of security and mastery of the system.

G. Prevention of erroneous tasks

A quality user interface should prevent users from performing an inappropriate or erroneous task in the first place rather than allowing the task to be performed and presenting a message afterwards saying that it couldn't be done. This can be accomplish by (a) disabling, or graying out certain elements under certain conditions, (b) providing checkboxes, dialog boxes, list boxes, scrolling lists, etc

H. Reliability

A quality user interface must be reliable or fault tolerant, that is, the ability
to deliver a minimum level of service in the face of hardware and software failures. The reliability R (t) of a system is a function of time, defined as the conditional probability that a system performs correctly throughout an interval of time [t_o, t], given that the system was performing correctly at time t_o. A friendly system design must allow for errors and at the same time provides easy paths back to recovery. The assumption should be that users would make mistakes. The ability to learn by trail and error is one of the strongest methods for learning.

I. Perform-ability

In many cases, it is possible to design a system that can continue to perform correctly after the occurrence of hardware failures and software errors, but the level of performance is somehow diminished. The perform-ability P (L, t) of a system is a function of time, defined as the probability that the system performance will be at or above some level L at the instant of time t.

J. Ease of evaluation

Following the execution of any task, users should be able to clearly and easily evaluate the effects of some actions. Incorporating mapping strategies (i.e., relationship between objects) is necessary to determine the effect of an action. For example, if a user pushes a button or slides a control, he or she should understand how that affects the system.

IV. APPROPRIATE FEATURES FOR USER SUPPORTIVE SYSTEMS

In the previous section we examined ten quality-considerations for the design of user-supportive interfaces. There are three basic features that are frequently proposed to address these considerations.[2]. These are: User modeling, Adaptation and Facilitation.

A. User Modeling

User modeling in the user support context refers to the computer- based construction and maintenance of individual user models. Ideally, a user model should be comparable to the user's internal mental model [2]. Users form an internal mental mode of themselves and of things that they are interacting with. Such models provide the user with explanation and possible predictive power for understanding his interactions. User models have been used to tailor system behavior and to provide computer- assisted instructions (i.e., tutoring, advising, coaching) and explanations. User models provide a sketch of individual users. They include:

i) *Parametric models*: This consists of a set of weighted or keyword attributes which are used as parameters to the task model (e.g. user profiles and alias facility). The attributes characterize the user (e.g., skill level), task (e.g., defaults), and preferences (e.g., how the task or interface behaves, what certain user- visible features should look like). These models are useful when the task is specific and well defined.

ii) *Discrete- event models:* This captures the intrinsic characteristics of how humans interact with a computer or a task. Such models support analysis of user behavior and provide behavior- to-structure transformations. Examples include the keystroke-level model, file-search profile, personalized telephone directories and accessibility for people with disabilities.

iii) *User knowledge models*: This can capture explicitly the descriptions of the users' knowledge of individual concepts (i.e., knowledge of the task, tools, task domain, and how the system interacts, and misconceptions). While the two previous models represent more general descriptors or behaviors, this type of model represents topics that the individuals have or need to acquire.

B. Adaptation

Adaptation is an attempt to bridge the gulfs between the system and the user's side or from the systems side. In the former, the user's behavior (possibly problematic) is collected to fit that required by the system through training, additional documentation, observations, inferences etc. In the latter, the system is modified so as to adapt to the user's needs and requirements. In this view, the system is the one at fault and the user is behaving sensibly. System adaptation efforts fall into one of two possible categories: adaptation of the system's design or the system's façade [8]. In the format approach, the system's design is adapted to fit the current needs and requirements of the user based on feedbacks from the user population about problems and limitations. This approach is more commonly known as rapid prototyping. This approach uses actual, untutored behavior to formulate the appropriate model behavior that the system should adopt. It makes the assumption that there is structure and consistency in user behavior. However, it is still concerned with constructing static interfaces for a prototypic user. Individualization is possible in the sense that a number of different interfaces may be constructed for a system. However, if the user population is large, this is not a viable route to individualized and variable interfaces. An alternative and more tractable approach is to adapt the system's façade according to individual needs and preferences [8], that is, the system's façade is custom-fitted to an individual. There are two basic approaches to this, namely: **adaptable** and **adaptive**. In the first (i.e. adaptable), the system's façade is adapted upon explicit request and know-how of the user. Through a set of tools (such as redo, undo, help) the user is able to tailor or customize his façade (see figure 6).

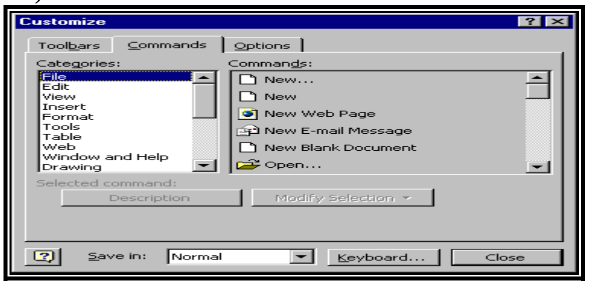


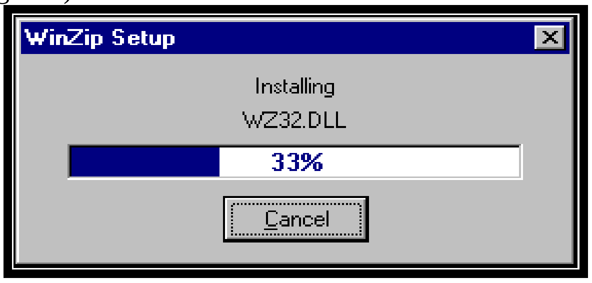
Fig. 6: Microsoft customizing facility

In the adaptive approach, an agent has been delegated the responsibility of when and how to carry out the necessary changes, occasionally interacting with the user [11]. By off-loading the low-level and operational decision-making for changes to the agent, it leaves the user free to concentrate on the higher-level conceptual or functional aspects of the problem at hand.

C. Facilitation

Facilitation is the act of making a process (either execution or evaluation of results) easier. Hence, we concentrate here on those techniques that facilitate both evaluation and execution of processes.

i) *Evaluation:* Feedback is important to the user for evaluating progress and completion of his task [5]. There are two types of feedback: user status feedback and system status feedback. User status feedback provides the user with feedback on all his actions. Examples include echoing the character typed, highlighting the object selected, and displaying error messages for mistakes. System status feedback provides feedback on the progress of the system's actions in processing the user's requests. A time-consuming operation may result in a period of delay. An anxious user may interpret such delay to mean that the system is hung, is in an infinite loop, or has ceased operation. Thus, intermittent feedback (e.g., percentage done indicators) does not only alleviate such anxieties but can show the progress of time-consuming operations (refer to figure 7).


WinZip Setup

Installing
WZ32.DLL

33%

Fig. 7: Percentage done indicator.

A good feedback or information delivery system may enable the user to have three questions conveniently answered at all times, regardless of applications. Where am I? What can I do? and how did I get here? Or where can I go? Or how do I get there?

ii) *Execution:* The first consideration here is the provision of tools and facilities that enable the user to deal with errors when they arise. Two examples that specifically address the reparation of errors are cited [12]. The first is the Undo and Redo facilities. They allow the user to recover from an error, to reverse an action, and to repeat actions (with possible modifications). The second is the Do What I Mean (**DWIM**) facility. DWIM provides an automatic spelling corrector to alleviate some of the operational difficulties associated with interacting with the system. The goal is to make the system insensitive to errors by providing an intermediate to correct such errors and thus minimize their impact on the user. The second consideration addresses the criticism that existing interfaces are unable to support both novice and expert users. In particular, experts prefer ways of quickly specifying what they want to do (i.e. with speed and power). The common facilities provided are accelerators and programmability. Accelerators enhance the speed with which a user may get things done. Examples include: function keys or keyboard commands, aliases, macros etc. Programmability enhances the user's power. Through a command language or a separate programming facility, the user has the flexibility and the means to modify, extend, contract, and chunk commands, enabling him to do the things that he wants, but which are not directly available in the basic repertoire of the commands. Finally, windows (i.e., partitioning the display into a number of virtual screens) are extremely useful and powerful as an extended external memory for the user [4,7,9]. They currently provide the best kind of support for concurrent activities and were the birth of the What You See Is What You Get (**WYSIWYG**) phenomenon. Windows and its features (icons, drag-and-drop manipulation of objects, etc) allows users to cache, organize, edit and view information directly so that valuable cognitive resources are conserved, and not occupied in remembering contextual information.

V. OPEN ISSUES AND PROBLEMS

This section identifies and discusses the open issues and problems with providing user support, such as lack of representation, lack of user information, lack of communication, lack of role and know-How, lack of consistent framework, lack of personalization.

A. Lack of Representation

To design a good user support system it is necessary to capture the evolutionary nature of users' behavior (i.e. dynamic models). However, user modeling is very open-ended because of the diversity of individuals, their knowledge, and their needs. It therefore, becomes combinatorial explosive to maintain all possible representations. An adaptive user interface solves this problem, by providing the capability to switch to an appropriate representation that best fits the particular situation.

B. Lack of User Information

There is no well-defined model to implicitly draw conclusions about the user based on observable behavior, the user's mental state, his intentions and goals. An additional benefit with having such rules is that it could provide the basis for successful plan monitoring and user assistance.

C. Lack of Communication

A major obstacle to achieving natural human-computer communication is the disparity between the interaction's requirements and the interpretive resources available to the participants. Specifically, we lack a crucial set of shared resources and information that implicitly guide conversation and allow participants to achieve mutual intelligibility. Existing systems do not account for the fact that the user's actions, often require interpretation in terms of a variety of situational factors that are, in turn under interpretation by the user. The flip side of the problem is that the system should also be able to obtain information about the user's intentions and goals based on his activities and the commands he issues. This means that the system should encourage the user, during the course of the dialogue to naturally externalize the knowledge and cognitive processing as the work is being done.

D. Lack of Role and Know-How

Lack of initiative (i.e., tool approach) would result in the user having to shoulder much of the burden of initiating and performing a task. A more realistic and desirable approach is to allow the computer, or more specifically the interface to act as the user's assistant, a lesser equal compared to the user. The assistant's contributions are in the form of compensating for the user's own limitations (e.g., cognitive), helping the user deal with menial and repetitive sorts of task, and dealing with simple housekeeping. Some have argued [1,5] that granting an assistant some initiative and some knowledge may work to the user's detriment. For example, the assistant may obstruct the user in trying to get his work done. To deal with this, the system can be designed to confer with the user in questionable situations. Some of the existing systems have implemented this approach using Help- assistance e.g. Microsoft word, Office professional etc.

E. Lack of Consistent Framework

A criticism against current efforts in windows user workspace is that many of the concepts (e.g. ***help, undo*** etc) have been introduced in an ad hoc and timid fashion. They do not follow a consistent framework and as such did not pervade throughout the system, with some being sporadically active. A framework is needed not only to ensure that the concepts pervade throughout the system and transcend particular applications but that the system supports new concepts.

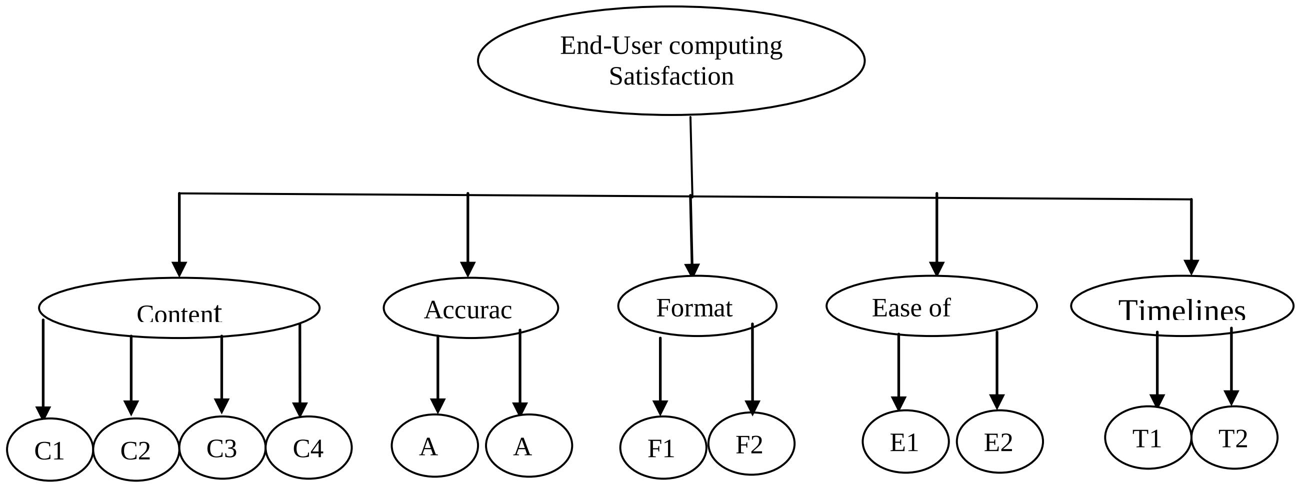
F. Lack of Personalization

In allowing users to have their own personalized behaviors and environments, (i.e. interface) the question that arises is how to let another individual use the same system and do reasonable work within it. Furthermore, there is the proliferation problem: variations of behaviors that may become too numerous to keep track of. There are issues of management and control of these behaviors. Both of these problems are very important but difficult to solve. Previous attempts have not addressed these problems adequately.

VI. MEASUREMENT OF USER SATISFACTION

User satisfactory feeling of a system can be evaluated by various criteria such as the degree of a user's satisfaction with the recommended item [13], user cognitive profile (CP), the amount of time a user spends to reach a decision and the decision errors that the consumer may have committed. Without real user's participation, the satisfaction of a consumer with the WUW is hard to measure. The classical effort-accuracy framework mainly investigated the relationship of decision accuracy and cognitive effort of processing information by different decision strategies in the off-line situation [13]. In this section we evaluate the WUW by explicitly measuring user satisfaction. In the WUW, the consumer first interacts with the system by inputting his or her preferences through the user interface. With the help of WUW, the system generates a set of recommendations to the consumer. This interactive process can be executed many times until the consumer is satisfied with the recommended results (i.e., a product to purchase) or gives up due to losing patience. We investigate user satisfaction as a measure of WUW performance. User satisfaction is an end product of post-purchase evaluation which is a posterior process where users evaluate the search tools in terms of whether the products they have found via the search tool are really what they want.

Reference [13] identified 12 items for measuring End User Computer Satisfaction (EUCS). The instrument comprises of five factors/components which are: content, accuracy, format, ease of use and timeliness. This study will adopt the procedures outlined by Doll and Torkzadeh to see if the model could be used to measure end user satisfaction derived from WUW. The 12-items are shown in the figure below while the questionnaire sample is listed in Apendix 1.



Key: C = Content, A = Accuracy, F = Format, E = Ease of Use, T = Timeliness

Fig. 8: EUCS Model Adopted for the study

VII. DATA AND DATA COLLECTION

This research uses the Doll and Torzadeh 12-item instrument to measure the level of satisfaction/dissatisfaction with WUW environment on end users {14}. The survey was conducted using 130 students who are currently undergoing a training course on ICT. After a practical lecture on WUW, the students were allowed to practice and use available tools on various WUW of choice. Hard copies of questionnaire were distributed to these students to complete. They were asked to answer the questions honestly and participation in the survey was anonymous. Only 108 of the questionnaires were useable.

VIII. SURVEY METHODOLOGY

The study was carried out using the same 12-item instrument of Doll and Torkzadeh and two global factors that measure the overall satisfaction and success serving as criterion. These two global factors introduced are: “Is the WUW successful?” and “Are you satisfied with the WUW?”. The five point Likert-type scale (1 = almost never; 2 = some of the time; 3 = about half of the time; 4 = most of the time and 5 = almost always) was used in the construct. Correlation between each item and total score of all items was tested; each item score to be tested was subtracted from the total before the test. A correlation of each item was conducted with the total of the rest 11 items remaining. We went further to examine the criterion related validity by testing the correlation between the total scores of the two global items measuring satisfaction and the total scores of the 12 items. The Principal Component analysis (PCA) was used to extract the components which were rotated using varimax to produce the loading values. The analysis was forced to have five factors so as to be in accordance with doll and Torkzadeh. All computations were done using SPSS software.

IX. RESULTS

Table 1, column 3 presents the results obtained in the correlation assessment between each item and total score of all items at 0.01 level of significance while Table 1, column 4 presents results of the correlation test between the two global items and the total score of items.

Table 1: Summary of Reliability and Criterion Validity measures of End-User Satisfaction

Item	**Description**	**Correlated item-to-total correlation**	**Correlation with Global Items**
C1	Does the WUW provide the precise information you need?	0.943	0.667
C2	Does the information content of the WUW meet your needs?	0.903	0.608
C3	Does the WUW provide reports that seem to be just about exactly what you need?	-0.158	0.020
C4	Does the WUW provide sufficient information?	0.675	0.553

A1	Is the WUW accurate?	0.801	0.576
A2	Are you satisfied with the accuracy of the WUW?	0.652	0.425
F1	Do you think the output is presented in a useful format?	0.732	0.564
F2	Is the information Clear?	-0.137	0.550
E1	Is the WUW user friendly?	-0.108	0.630
E2	Is the WUW easy to use?	-0.094	0.575
T1	Do you get the information you need on time?	0.806	0.601
T2	Does the WUW provide up-to-date information?	0.880	0.601

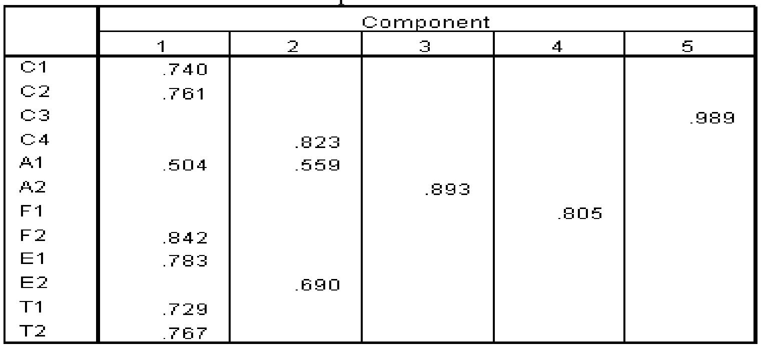
For the reliability test, Doll and Torkzadeh did not stipulate any cut-off threshold standard but following their example of 0.5 cut-off value, we discover that four of the items (C3, E1, E2 and F2) did not reach the cut-off value. On the other hand, the criterion validity test shows that one item (C3) falls short of the cut-off mark of 0.4. Due to the results of the experiment, all items were considered to be valid apart from items C3, E1,E2 and F2. In order to ascertain if a Factor analysis could be carried out on the data set, Barlett's test of sphericity and Kaiser-Meyer-Olkin Measure of Sampling Adequacy was carried out.

Table 2: A Priori Test before Factor Analysis

Kaiser-Meyer-Olkin Measure of Sampling Adequacy		.892
Bartlett's Test of Sphericity	Approx. Chi-square	1.341E3
	df	66
	Sig.	.000

From Table 2 above, Barlett's test of sphericity is significant and the KMO value of (.892) is also large enough, so a factor analysis on the items could be carried out.
In conducting the Factor analysis, items with factor loading less than 0.5 were deleted. Table 3 below summarizes the results.

Table 3: Rotated Component Matrix of the 12-Items



	Component				
	1	2	3	4	5
C1	.740				
C2	.761				
C3					.989
C4		.823			
A1	.504	.559			
A2			.893		
F1				.805	
F2	.842				
E1	.783				
E2		.690			
T1	.729				
T2	.767				

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.

a. Rotation converged in 7 iterations.

Using a threshold value of 0.7, it could be seen that items C1, C2, F2, E1,T1 and T2 are highly correlated with factor 1, C3 is highly correlated with factor 5, C4 with factor 2, A2 with factor 3, F1 with factor 4. A1 and E2 did not reach the threshold value.

X. CONCLUSION

A host of quality considerations that addressed user support in windows environment is presented as well as user satisfaction rating. In allowing users to have their own personalized behaviors and environments, the question that arises is how to let another individual use the same system and do reasonable work within it. Furthermore, variations of behaviors may become too numerous to keep track of. There are issues of management and control of these behaviors. Both of these problems are very important but difficult to solve. Previous attempts have not addressed these problems adequately. In solving these problems, an experiment was conducted using an existing EUCS model. Was validated on ten out of the 12-items suggested thus giving room for more research and development of user support tools and facilities to cater for different varieties of users. Thus, we look forward to many exciting new user-friendly interface research results in the future.

REFERENCES

[1] Norman, D. A, *Cognitive Engineering,* In User-Centered System Design: New Perspectives on Human-Computer Interactions, ed. D.
A Norman and S. W. Draper, Lawrence Erlbaum Associations. Pp. 31-61, 1996.

[2] Lee, A., *UserSupport: Considerations, Features and Issues In office and Database Research. '87',* ed. F.H. Lochovsky, pp.1-2, 1987.

[3] Nikerson, R. S and Landauer, T. K. *Human- Computer Interaction: Background and Issues*. In M.G. Helander, T. K. Landauer& P. V.
Prabhu (Eds) Handbook of Human- Computer Interactions, Elsevier, Amsterdam, pp.3-32, 1997.

[4] Head, A. J., and Bjorner, S, *A Question of Interface Design: How Do Online Service GUIs Measure Up,*? Online: 21 (3), pp.20-24, 1997.

[5] Bakpo, F.S, *Intelligent Software Agents: Concepts and Application in modern Information Retrieval System,* NCS Conference
Proceedings, vol 14, pp 295-307. 2003.

[6] Mayers, B.A. and Rosson, M.B, *Survey on User Interface Programming, Proceedings* SIGCHI '92: Human Factors on Computing Systems. Monterrey, C.A. May 3-7; pp195-20, 1992.

[7] Bakpo, F.S, *Computer use and Applications,* Godjiksons Publishers, Nsukka, 128pp, 2002.

[8] Rissland, E. L *Ingredients of Intelligent User Interfaces,* International Journal of Man- Machine Studies: 21 (4), pp. 377-388, 1984.

[9] Ogden, W. C. and Bernick, P, *Using Natural Language Interfaces,* In M.G. Helander, T. K. Landauer & P. V. Prabhu (Eds), Handbook of Human-Computer Interactions, Elsevier, Amsterdam, pp.137-163, 1997.

[10] Sleeman, D, *UMFE: A User Modeling Front- End Subsystem.*, International Journal of Man- Machine Studies: 23 (1), pp. 77-8, 1985.

[11] Shneiderman, B. (1997) " Direct Manipulation for Comprehensible, Predictable and Controllable User Interfaces." *Proceedings of the 1997 International Conference on Intelligent User Interfaces. Retrieved from the web address: http://www.acm.org/pubs/citations/proceedings/uist/238218/p33-shneiderman/*

[12] Vitter, J. S, *US&R: A New Framework for Redoing, IEEE Software:* 1(4), pp. 39-52, 1984

[13] Zhang, J. and Pu, P, *Effort and Accuracy Analysis of Choice Strategies for Electronic Product Catalogs*, In Proceedings of 20th ACM Symposium on Applied Computing (SAC-2005) Santa Fe, New Mexico PP.808-814, 2006.

[14] Doll, W. J. and G. Torkzadeh, *The Measurement of End-User Computing Satisfaction,* MIS Quarterly, Vol. 12, No. 2:259-274, June 1988.

APPENDIX 1

Modified Sample Questions for the survey using Doll and Torkzadeh's EUCS instrument

G1: Is the WUW successful? G2: Are you satisfied with the WUW?.
C1: Does the WUW provide the precise information you need?
C2: Does the information content of the WUW meet your needs?
C3: Does the WUW provide reports that seem to be just about exactly what you need?
C4: Does the WUW provide sufficient information?
A1: Is the WUW accurate?
A2: Are you satisfied with the accuracy of the WUW?
F1: Do you think the output is presented in a useful format?
F2: Is the information Clear?
E1: Is the WUW user friendly?
E2: Is the WUW easy to use?
T1: Do you get the information you need on time?
T2: Does the WUW provide up-to-date information?