Models and methods of designing human-machine interaction-oriented interfaces

Ozod Radjabov  
*Tashkent university of information technologies (TUIT), Uzbekistan, uznis2015@gmail.com*

Shakhrillo Bobokulov  
*Tashkent university of information technologies, (TUIT) UZBEKISTAN, boboqulovshax@gmail.com*

Hojiyev Sunatullo

Behruz Boboqulov  
boboqulovbehruz1995@gmail.com

Follow this and additional works at: [https://uzjournals.edu.uz/tuitmct](https://uzjournals.edu.uz/tuitmct)

Part of the Computer and Systems Architecture Commons, Digital Circuits Commons, and the Other Computer Engineering Commons

**Recommended Citation**  
Available at: [https://uzjournals.edu.uz/tuitmct/vol3/iss1/5](https://uzjournals.edu.uz/tuitmct/vol3/iss1/5)

This Article is brought to you for free and open access by 2030 Uzbekistan Research Online. It has been accepted for inclusion in Bulletin of TUIT: Management and Communication Technologies by an authorized editor of 2030 Uzbekistan Research Online. For more information, please contact sh.erknov@edu.uz.
Models and methods of designing human-machine interaction-oriented interfaces

Cover Page Footnote

This article is available in Bulletin of TUIT: Management and Communication Technologies: https://uzjournals.edu.uz/tuitmct/vol3/iss1/5
Models and methods of designing human-machine interaction-oriented interfaces

Ozod Radjabov, Shakhrillo Bobokulov, Hojiyev Sunatullo, Boboqulov Behruz

Abstract: Formalization approaches of user interface design (UID) in conjunction with model driven techniques aim to improve the usability in terms of conformity to standards or style guides and to leverage code generation of interactive software systems, so that various UI platforms for web, desktop or mobile Applications are supported. Because large parts of the UI are described platform independent instead of platform dependent implementations, re-usability of the UI concept is also improved. However, UI formalization requires the usage of a formal UI description language and a higher level of abstractness compared to concrete UI code. These languages need to be learned by the UI designer. In practice, most parts of a user interface are still manually designed and coded individually for every platform. This paper describes how HCI (Human Computer Interface) patterns that are described formally can be used in conjunction with model-based user interface design in order to make it easier for the designer to use formalization techniques for the development of user interfaces. The approach uses two UML profiles: The MBUID (Model-Based User Interface Design) profile and the HCI pattern profile. With these profiles formal models of interactive systems can be created on a platform independent level. The user interface is then automatically generated by model-driven development tool chain.

Key words: MDA, user Interface, usability, UML, UML profile, PIM, PSM, transformation, user interface code generation, model driven architecture, model based user interface design, human computer interaction, usability pattern,

1. Introduction

Usability and software ergonomics play an important role in software product quality and success today. While software engineering has lead to structured approaches and processes for improving quality, usability issues and user interface design are often still regarded as rather optional or late-phase activities. A reason for this gap is that user interface design is either executed ad-hoc and unmanaged by software developers during application development or as a separate activity with only loose coupling to other elements of the software development process.

Human-machine interaction interface is the medium between human and machine to transmit information and is the specific form of expression between the human, machine, and environment, which is also the necessary means to realize the interaction. In
the model of human-machine system, through the visual and auditory organ, people receive information from machine; then, with the processing and decision in brain, the locomotive organ reacts to realize the information transmission between human and machine. On the other hand, all sorts of machine’s display have effect on people, which will realize the information transmission between machine and human.

The layout problem of human-machine interaction interface was getting more and more inseparable with artificial intelligence technology. Researchers began to pay attention to various computational intelligence algorithms, such as genetic algorithm, particle swarm optimization algorithm, simulated annealing algorithm, ant colony algorithm, and tabu search. Besides, according to the NO Free Lunch theorem proposed by Professors Wolpert and Macready in Stanford University [1], a single optimization algorithm had its advantages and disadvantages of application. From the perspective of solving optimization problem, the fusion of different types of algorithmic mechanism and giving full play to their respective advantages were the inevitable development trend to solve the problem. Hybrid intelligent optimization algorithm had become an important strategy to solve practical engineering problems [2].

The effect of the algorithm in different applications was different, so, according to the characteristics of the cabin, solving algorithm should be suitable for the layout problem. The current layout research was focused on the improvement of space utilization [3], so the traditional optimization objective was difficult to meet the requirements of human body’s comfort in operating. Human physiological and psychological characteristics should be considered in the layout design, so as to make the operators have comfortable operating posture. Thus, from the perspective of cognitive psychology [4], the layout principles of human-machine interaction interface were summarized, and the human cognitive characteristics were quantified as the layout constraints. GA-ACA [5] would be applied to realize the intelligent layout optimization of human-machine interaction interface of cabin.

2. Implementation Method and Model

An extensive model landscape has been developed in the last decades for analysis and design in software development with regards to user-oriented approaches. This section provides an overview necessary to understand the current model and model integration discussion.

2.1. HCI Models for Task Analysis and Modeling

The goal of finding a modeling technique that serves for task analysis and modeling has produced a variety of different techniques in the human-computer interaction domain, ranging from psychological to rather technical methods.

Some of them are described in the following to give an impression of the wide range and provide hints for developers and research projects that may have special
requirements pointing to one of those methods.

ETIT analysis assumes that users have to map intern tasks (IT) to extern tasks (ET). Using a matrix it is possible to estimate the effort for knowledge transfer for similar tasks.

GOMS (goals, operators, methods, selection rules) takes over the notion of goal, sub-goal, operator method and selection rule from the problem solving theory. Goals are subdivided into sub-goals; activities arise from the application of operators, which transform to methods – i.e. high-order operators – by control structures.

CLG (common language grammars) are better suited for command based systems. It differentiates a task layer with extern tasks, a semantic layer with intern tasks, a syntactic layer with concepts and commands and an interaction layer with actions and presentation.

TAGs (task action grammars) do not have this focus on commands, which makes them suitable for other user interfaces, too. They provide a meta language for the description of task description languages. A tag model contains task elements (features), a list of tasks and dedicated elements and substitution rules (grammar) for the production of activities.

CCT (cognitive complexity theory) helps determining the cognitive complexity of a system by judging the knowledge necessary. CCT has been developed from GOMS by projection of the procedural descriptions to conditional and action rules.

These models target behavioral modeling, which is more or less procedural, and often have no relationship to software engineering models. As most of these models are completely incompatible with modern object-oriented software development approaches, they are often used by a small group of specialists only, which makes the models only rarely useful as a model basis for software development projects and integration with software engineering. With the rise of UML (OMG, 2007; OMG, 2007b) use cases in a combined graphical and textual form have been introduced to software engineering practice as black box modeling concepts.

2.2. Establish the Cognitive Model of Human-Machine Interaction Interface

Human-machine interaction process is actually a process of information processing. Through the characteristics about mental labor in cognitive psychology, such as the research of memory, understanding, and communication, the designed human-machine interaction interface tries to reduce people’s cognitive burden as much as possible, making the product easy to learn, easy to use, and with high efficiency. The idea of human-machine interaction interface layout design was to build cognitive model based on the information processing mechanism. Then, on the basis of people’s thinking characteristics, using the rule of information organization, visual search pattern, and memory characteristic, the human-machine interaction interface layout would conform to operators’ cognitive ability for interface information.

There were plenty of studies on cognitive processes; psychologists’ emphasis on the study of cognitive processes was different at different periods. Information processing model described the main elements or stages in the human information processing.
processing and the hypothetical relationship between them. Most of the models were consistent with this basic framework. And on this basis Wickens et al. [6] put forward the model of information processing with the attention function. Sun et al. [18] put forward cognitive synthetic model, which brought cognitive process into the interaction between human and environment system, according to the thought of parallel processing of layered information, and output by competition and coordination. Taking and integrating the advantages of the predecessor’s research as reference, this paper put forward the cognitive model of human-machine interaction interface of cabin (shown in Figure 2).

Figure 2. The cognitive model of human-machine interaction interface of cabin.

As can be seen from Figure 2, cognitive processes are placed in the interaction system of human, machine, and environment. Through visual and auditory organ, the operators observe the system’s operating situation and proceed with sensory processing. Combining call rules and knowledge in long-term memory and call targets and tasks in short-term memory with constraint module, the sensory information processing is conducted. Information processing includes intuition layer, template layer, reasoning layer, and comparator. Information is parallel-processed by the three layers of intuition, template, and reasoning, and the schemes are produced in the competition and cooperation in the three layers and output in the comparator. Finally, the selected corresponding method is carried out. Information acquisition, processing, and performing all need to interact with the constraint module, which is the operators’ subjective understanding, habits, and principle of information processing. Perception, decision-making, and response implementation all need to consume attention. In order to obtain the best cognitive ergonomics, appropriate cognitive strategies should be adopted to balance the quality and speed of the information processing in cognitive system. This model could describe the operators’ cognitive process clearly, so it is suitable for application in the layout design of human-machine interaction interface of cabin.

2.3. Object-Oriented Methods

Ozod Radjabov, Shakhrillo Bobokulov, Hojiyev Sunatullo, Boboqulov Behruz

2020, 2 (45)
With object-oriented paradigms becoming popular in the field of programming languages under the notion of object-oriented programming (OOP), modeling concepts also have turned in this direction to keep paradigms of models and code compatible.

In the early nineties, for example OBA (Rubin & Goldberg, 1992), OOA (Coad & Yourdon, 1991) and OOA&D (Martin & Odell, 1992) have been published, all aiming at an object-oriented analysis and providing a basis for the following object-oriented design. (Janssen et al., 1993)

The concepts of OMT (Rumbaugh et al., 1991) and OOSE (Jacobson et al., 1992) were integrated with the approach of Booch to form todays industrial quasi-standard Unified Modeling Language (UML; Scott, 2004) for object-oriented analysis and design, while the OPEN modeling language (OML; Firesmith et al., 1998) has been developed as a sort of competitive language to UML. A comparison of both languages from the perspective of OML can be found in (Henderson-Sellers & Firesmith, 1999), where it is argued that UML can only be considered as hybrid object-oriented and is strongly focused on C++ implementations.

Although, with the Meta Object Facility (MOF; OMG, 2002), OMG strongly pushes UML into the direction of model integration, which will lay the fundament for emerging integrated modeling approaches like the one presented in this chapter.

With the introduction of semantic web technologies, the semantics of models and their (semi-)automated interpretation / inference have become more important leading to more powerful models, which still focus mainly on structural aspects like data structures, types and dependencies.

3. An Interaction-modeling Approach

For the analysis and requirements phase, pure object-oriented models like OOA are too far away from the mental models of non-developing stakeholders like users and customers and are too static to describe the processes rather than pure entities in requirements gathering. This is the reason why more narrative and informal representations like use cases have gained importance, which may still be embedded into an object-oriented framework – like UML use cases – but are informal enough to let all stakeholders participate in the creation process.

While UML use cases (Jacobson et al., 1992) are defined as a representation of a systems view (Rumbaugh et al., 1999), task cases – also called task descriptions (Rumbaugh et al., 1999) or essential use cases (Biddle et al., 2001) – have been developed to allow for easier expression and focus more on the user and his/her intentions. Starting with this simple but impressive modeling technique, users and customers can join the analysis process, because they are able to contribute in their own words and with natural language concepts.

This trend shows that flow-oriented, natural language based and semi-structured description methods yield high potential for...
successful requirements analysis and negotiation to software development projects.

Due to different aspects and people working with task cases and use cases, these two – originally very similar – techniques are often used in parallel, even in modern approaches for software engineering and usability integration, like usage-centered design.

It is also possible to refine, expand and integrate essential use cases into system use cases. Merging both concepts together with the ability of stepwise refinement (top down) or – the other way round – construction from almost atomic interactions (bottom up) provides a sustainable concept for interaction modeling. Thus, development projects are no longer forced to develop two or more sparsely connected models in parallel for interaction flow description.

### 3.1. Layout Design of Driller Control Room on Drilling Rig

Take the layout design of human–machine interaction interface of driller control room on drilling rig as an example to illustrate the proposed method. The 16 manipulators of ZJ120/9000DB rig console would be arranged (the grey area as shown in Figure 2). First of all, the 16 manipulators need to be encoded. As shown in Table 1, when encoding, layout Principle 1 needs to be reflected; that is, manipulators with similar function will be arranged in one area, to reduce the search time for operators.

<table>
<thead>
<tr>
<th>#</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Partition coding for manipulators

It is necessary to simplify the layout area and layout objects when describing the mathematical model, to facilitate a digital description of design variables. According to the human upper limb dimension to determine the layout space and simplify the layout area as rectangular area (450 × 400 mm), there are three types of manipulators, including buttons, rotary knobs, and switch knobs. Their sizes are slightly different, and the specific sizes are shown in Table 2. The spaces between manipulators are set as equal, and the interval is 100 mm. After being simplified, the layout of manipulators can be regarded as a scheduling problem, first using GA-ACA to find a good sorting and then according to the order to establish the actual layout.

<table>
<thead>
<tr>
<th>#</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Partition coding for manipulators
Figure 3. Layout schematic diagram of ZJ120/9000DB rig console (before optimization).

Drilling process is mainly divided into three working conditions: pull out of hole, run in hole, and normal drilling. Considering the principle of operating sequence in multiple working conditions, determine the manipulators’ operating sequence and calculate each manipulator’s weight relative to Principle 2. Through questionnaires to consult drillers, determine the relative judgment matrix of the manipulators’ importance and frequency and calculate each manipulator’s weight relative to Principle 3. Analytic hierarchy process will be used to calculate each manipulator’s weight of and , and the calculation result is shown in Table 3. Finally, determine the relative correlation between the manipulators. The correlation between manipulator and itself is expressed with 1, and the correlation between the manipulator and other manipulators is expressed in decimal within 0~1. The greater the correlation between manipulators, the greater the correlation value. The correlation between the 16 manipulators is shown in Table 4.
Table 3. The manipulators’ weight relative to Principles 2 and 3.

| 1 | 2 | 3 | 4 |

4. Optimization Result

After completing the above data preparation, as shown in Figure 4, the operating parameters of GA and ACA need to be set up before the computer aided optimization calculation proceeds. First, set the control parameters of GA: population size N=50, crossover rate Pr = 0.6, mutation rate Pm = 0.2, and number of partition S = 4. Second, set the control parameters of ACA. The amount of ants is m = 10, a =1, B = 1, and p = 0.5.

In the system of GA-ACA calculation module, set the end condition of GA: minimum genetic iterations GENmin = 15, maximum genetic iterations GENmax = 50, minimum evolution rate GENEp = 3%, and GENE1=3. Set the control parameters of pheromone: Q=500, Tmin=10, and Tmax=100. The optimal 10% of individuals in the last generation in GA would be selected as the solution set of genetic optimization, which would be transformed into pheromone values. If manipulator was adjacent to manipulator j in genetic optimization solution, the pheromone added 10 on the path (i, j), and all the initial values of pheromones should be set like this. When meeting one of the following conditions, the ACA should stop. The number of iterations reaches ANTmax=50.(2) For continuous three generations in the iteration, the improvement rate of offspring optimization is less than 0.5%.

Click the button of algorithm running on the system interface, the system calculates via the program, after 50 times genetic iterations and 11 times ant colony optimization iteration; the value of the optimal layout scheme is 2.822. The sequence of the manipulators is obtained, that is, 4, 1, 3, 2, 6, 5, 7, 8, 9, 10, 12, 11, 14, 13, 15, and 16. According to the optimized sequence, the layout scheme is shown in Figure 4.
5. Generation of other Final Artifacts

So far, we have only discussed the ability to generate the final artifacts.

We define final artifacts as all artifacts that experience no further transformation and are therefore in their final state – unlike source models or intermediate representations.

User interface prototype, help and documentation from IML models. But a fully staffed IML model provides enough information to allow for generation of artifacts beyond GUI models.
Figure 5. The generation process also provides final artifacts different from generation results.

Once the IML model has been completed, black box test cases for application testers can be generated from the information given in the interaction flows and attached data definitions.

The advantage of these and other generateable artifacts is that changes in the IML model have direct impact on them. Generating artifacts from an IML model, it is not necessary to find and track changes on every artifact but only on the IML model. Using the IML approach for generation of the artifacts as shown in figure 5 will render possible the application of IML as an integrated base model for user-oriented software development processes.

6. Conclusion Based on the theory of cognitive psychology, according to Wickens information processing model, the cognitive model of human-machine interaction interface was established. Considering people’s information organization law, visual search law, user memory characteristics, and so on, the human-machine interaction interface should conform to operators’ cognitive ability for interface information. Based on the cognitive model, the layout principles of human-machine interaction interface were summarized as the layout constraints. The problem of solving layout scheme was

Ozod Radjabov, Shakhirillo Bobokulov, Hojiyev Sunatullo, Boboqulov Behruz
2020, 2 (45)
transformed into combinatorial optimization problem, and GA-ACA was put forward to solve the problem, which realized the algorithmization of artificial optimization process.

Taking the layout of manipulators as example, the objective function of layout optimization was built according to each principle. Use GA to generate the solution of layout scheme, which would be transformed into initial pheromone as ACA required. Taking the difference value of comprehensive weight of the manipulators for the layout principles as heuristic information of ACA, combined with the pheromone provided by GA, the positive feedback optimization mechanism of ACA was used to solve further. GA-ACA integrated the complementary advantages of GA and ACA, which had good optimizing performance and time performance and improved the design efficiency. Taking the 16 manipulators of ZJ120/9000DB rig console as layout example, layout design system of human-computer interaction interface for cabin was developed by Visual Basic, which validated the above layout optimization method.

Current developments in software engineering and business IT show a trend towards regarding applications not as fixed and local installations. Instead, a process towards virtualization and decentralization takes place for computing and data as well as business processes and especially services – often referred to as “cloud computing”. With dynamically changing services, processes and compositions in decentralized runtime architectures, design-time modeling and generation will not be sufficient anymore.

References

18. Eclipse Foundation: www.eclipse.org