Towards Model-based Usability Evaluation of Interactive Application: Detecting Unexpected Situations and Validating System Task Model

Mouna Jarraya, Faouzi Moussa
National School of Computer Sciences, Cristal Laboratory, University of Manouba
2010 Manouba, Tunisia
e-mails: {jarrayamouna, faouzimoussa}@gmail.com

***Abstract*—Usability evaluation is not fundamental only in the interactive application development process, but also at operation time. In this paper, we discuss how a task model could be used to detect any incompleteness due to poor requirements when designing an interactive application. The focus is on the comparison between user task and user activity to detect deviations in order to mend the task model. It is about detecting unexpected situations and validating the task model as it brings many benefits (e.g., it represents a user documentation; it is used for training operators, etc.). Indeed, it aims to ensure the stability and effectiveness of the system when context change of the environment system occurs.**

Keywords-Model-Based Usability Evaluation; Task Model; Petri Net; Activity Model; Human Computer Interaction; Design Validation; Unexpected Situation.

I. INTRODUCTION

The unexpected in the context of application development is defined as anytime when human operators deviate from the "linear" norm [1]. It happens when we define poor requirements throughout the design process. Literature exposes many approaches that use different techniques to detect erroneous actions and unexpected events [5].

In this work, we focus especially on how a task model enables us to evaluate and validate the effectiveness of an existing interactive application by identifying which tasks are supported by it and which ones are not. We propose a model-based approach to support usability evaluation involving users.

Task models are artifacts where we gather all the user requirements for the tasks to achieve a goal. They are usually used at the design and development process of interactive applications. However, using task models at operation time (when the interactive application has been deployed and used) brings many benefits such as users training, helps designers to evaluate their product, etc.

The remainder of the paper is structured as follows. Section II presents the state of art, gives task model and activity model definitions and it provides a quick overview of related work on how task models could be useful to improve usability of interactive applications which are already deployed and used. Section III presents some notions considered as corner stones in the development of our approach and gives an overview of it. The last section summarizes the contributions of the paper and highlights future work.

II. STATE OF THE ART

Few works addressed the issues of using task models in usability evaluation. In this section, some seminal definitions and concepts are given and we expose related work.

A. Definitions

Usability evaluation: The goal of usability evaluation is to find out possible usability problems of a system and discover ways to resolve them [10].

Likewise, the International Organization for Standardization (ISO) defined usability as the "Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [12]. In order to measure how usable a system is, the ISO standard defines three attributes:

- Effectiveness: Accuracy and completeness with which users achieve specified goals;
- Efficiency: Resources expended in relation to the accuracy and completeness with which users achieve goals;
- Satisfaction: Freedom from discomfort, and positive attitudes towards the use of the product

Many existing models for usability are present in literature, for example, Nielsen's model [13], ISO, etc.

Usability evaluation methods can be classified as [14]:

- Model-based approaches where a significant model related to the interactive application is used to drive the evaluation. Various types of models, such as user, context and task models have proved to be useful in the design and evaluation of interactive applications;
- Inspection-based assessment, where some expert evaluates the system, or some representation of it, according to a set of criteria;
- Empirical testing where direct use of the system is considered.

Task model: Task analysis typically results in a task model, which is a description of any interactive task to be accomplished by the user of an application through the application's UI (User Interface). Individual elements in the task model represent specific actions that the user may

 ISBN: 978-1-61208-468-8

undertake. Information regarding subtask ordering as well as conditions on task execution is also included in this model [8].

Activity model: We can characterize the activity of a human or a system who/that has to execute a task [1]. It represents actions that the user undertakes to accomplish a task. An activity model must be developed through observations and interactions with users [11]. It results from the execution of the task model. Thus, an activity model is a response to a task model.

B. Related Work

Several studies used task models as a tool to design interactive applications. Nevertheless, less works paid attention to their use to evaluate and validate Human-Computer interaction (HCI).

Paternò et al. [8] performs a systematic inspection-based analysis to improve both usability and safety aspects of an application. It aims to evaluate what could happen when interactions and behaviors occur differently from what the system designer assumed. It indicates a set of predefined classes of deviations that are identified by guidewords. These latter are used to analyze deviations by interdisciplinary groups. The main drawbacks are that it costs in terms of time and effort.

Another model-based approach incorporates human error analysis with task modeling [9] introducing the concept of Error Pattern. Error Patterns are prototypical deviations from abstract task models, expressed in a formal way by a model transformation. Nonetheless, task model, in this approach, is used in the design process and not at operation time.

Some authors combine different techniques including formal analysis of models, simulation and, in particular, analysis of log data in a model-based environment [7]. They extract test scenarios from the task models which will be simulated and executed to finally produce based-model logs. Thus, these logs support the assessment of the models, and allow iteration while performance of these techniques does not meet the corresponding requirements. Nevertheless, designer has to locate by himself where changes have to be made and additionally how to make them.

Martinie et al. [6] extends the previous approach and proposes a framework for connecting task models to an existing, executable, interactive application. It develops a co-execution environment that enables a systematic correspondence between the user interface elements of the interactive application and the low level tasks in the task model. It allows users to benefit from information available in the task model while interacting with the actual system such as assessing work complexity, operators' workload, identifying areas for improvement, etc.

III. OUR APPROACH/CONTRIBUTION

In the following, the HCI lifecycle is detailed to better understand how general activity model could be extracted. It is important to figure out problems and conflicts of the model and to follow up a model-based resolution as explained below [4]. This model has been used and depicted in our approach.

A. HCI Lifecycle

Abed et al. define the lifecycle of specification and generation of context-aware interfaces in two parts: design and implementation [2] (see Figure 1).

Human-Computer System analysis represents the identification of important subsystems in order to define their behavior. The objective is to identify the different graphics views appropriate for each subsystem.

This step will allow us to identify for each subsystem, the appropriate user's tasks. Task analysis helps to define the actions relating to the user's tasks.

Indeed, modeling the user's task has a strong impact on the design of user interface.

It allows us to deduce the user's requirements for the user's interface specification. The latter enables us to identify a set of graphic interface components for each user's task. Consequently, HCI is automatically generated.

At this stage, we define the experimental protocols for our Human-Computer System to simulate at real-time in order to detect some design errors or to identify situations not already expected by designers. For each simulation, we extract the user's cognitive activity and verify that it matches with user's task model or not. We iterate the cognitive activity analysis process until there would be no mismatch between the real activity and the designed task. We talk about validation of our HCI and we obtain the general activity model.

B. Model-based resolution

A task model represents a hierarchy of tasks. A task is a goal and a procedure that is a set of sub-tasks having temporal and composition relationship. Elementary tasks are only decomposable into physical actions. In the human decision theory of the operator when solving a problem in, skill-based behavior represents sensory-motor performance during acts or activities [4]. It is based on simple

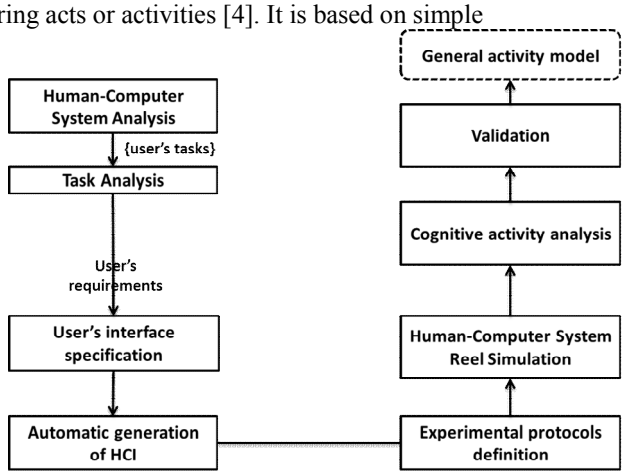

Figure 1. HCI lifecycle according to Abed and Millot

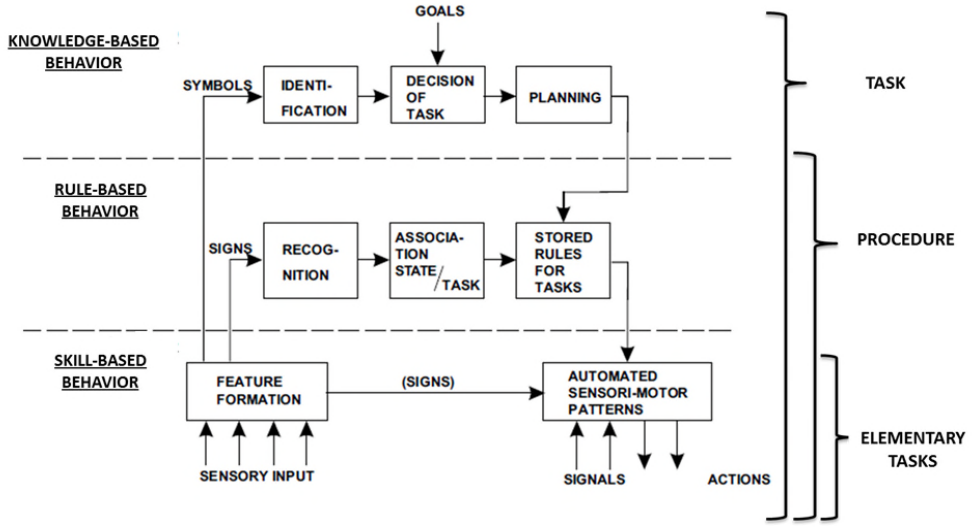

Figure 2. Simplified illustration of three level of performance of skilled human operators [4] and their correspondence with task's elements

feedback control. It corresponds to the elementary tasks level (see Figure 2).

At the next level of rule-based behavior, the composition of such sequence of subroutines in a familiar work situation is typically controlled by a stored rule or procedure. It can be related to task's procedure that is a sequence of sub-tasks controlled by a set of rules.

For unfamiliar situations with no rules for control available from previous encounters (in rule-based behavior rule), the control of performance must move to a higher conceptual level, in which activity is knowledge-based and goal-controlled. Based on model-based resolution, elementary tasks would be the starting point of the task analysis and subsequently the task model.

C. *Petri Nets modeling language for elementary tasks*

Several task modeling approaches and techniques are available. However, it is mandatory for us to choose the most suitable modeling approach. Riahi et al. [3] depict a comparative study between different models (e.g., key-value model, ontology, etc.). They find multiple proofs of the Petri Net's ability to model tasks.

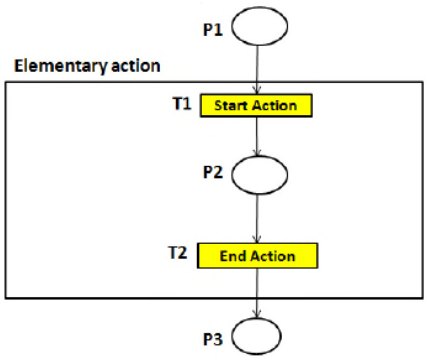

Figure 3. Structure of an Elementary action [3]

Modeling with Petri Nets (PN) enables to do formal verification and prior validation of tasks, which would save a considerable amount of time in the development cycle of the interface [3].

There is a set of criteria that convinced us to adopt Petri Nets in our work as presented in [3]. We considered that the user task is composed of sets of elementary PN structures. The modeling of an elementary structure is illustrated in Figure 3. All the users' actions (elementary or composed) are organized according to typical compositions: sequential parallel, alternative, choice, iterative or of-closure. The principles of only two compositions are described below.

Sequential composition: The composition of "n" sequential actions reflects the sequence of their execution. The sequential composition of n actions is ensured by combining the place "end" modeling the action i, with place "begin" of the action i + 1 (see Figure 4). In the example, the user performs three actions one after the other.

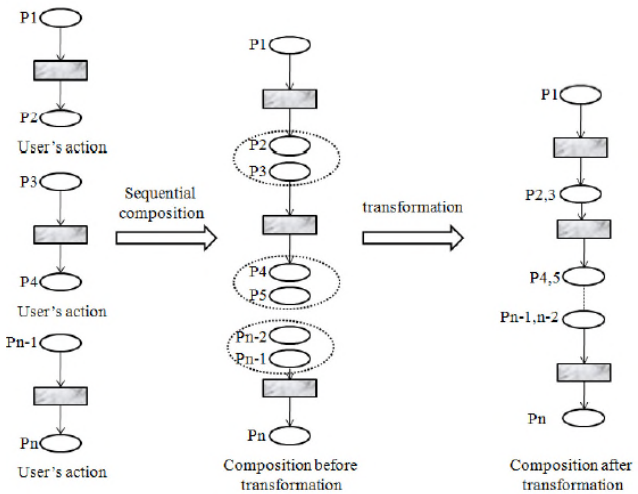

Figure 4. Sequential composition [3]

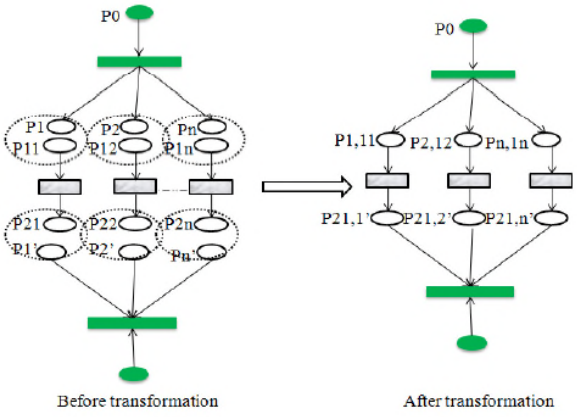

Figure 5. Parallel composition [3]

Such an interaction will be modeled by a sequential composition of three elementary structures (modeling three elementary actions).

Parallel composition: Parallel composition expresses the possibility of simultaneous execution. Parallelism is ensured thanks to an input synchronization place. This place activates at the same time all the places of initialization of the parallel actions to be executed. Ensure the parallel composition of actions; it is necessary to synchronize the places of entry and those of exit of those actions (see Figure 5).

D. Proposed approach: an overview

As seen above, the proposed approach aims to detect any deviation from what task model assumes comparing to interactive application's activities. For this matter, user activities would be compared to a set of elementary tasks in order to compute or to calculate a Δ. The latter would identify what, when and where dissonance or deviation occurred. If it is the case, mending the task model would be necessary (see Figure 6).

From this process, three major components are highlighted:

Computing Δ: This step requires as inputs user tasks (from task model) and user activities. For the latter, a correspondence controller traces the user actions or activities on the interface via Petri Nets mechanism mentioned above.

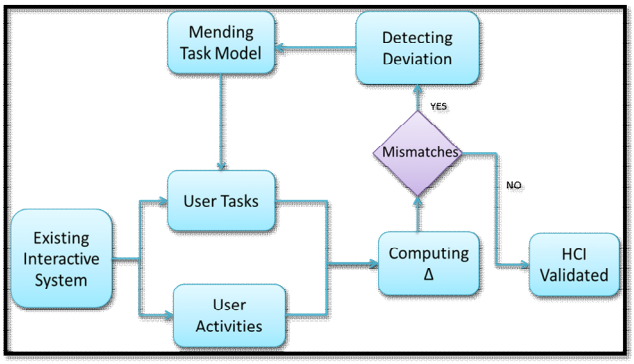

Figure 6. Usability evaluation process

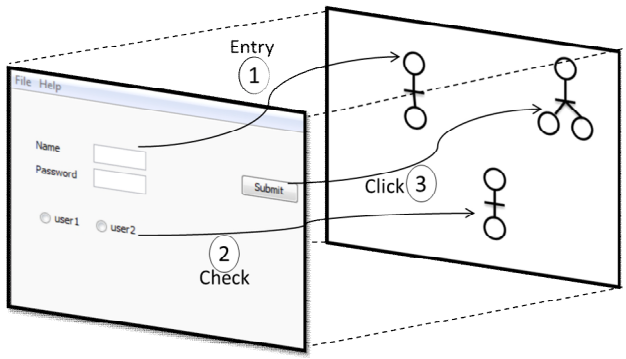

Figure 7. Correspondence Controller

In fact, for each user action, an elementary Petri Net is associated to it (see Figure 7).

After that, assembling elementary Petri Net will follow the sequence of user actions until obtaining the user activity and finally an activity model for a high-level abstraction.

Thus, user tasks and user activities are modeled in the same way as they reference to elementary actions (elementary Petri Net). They are considered now as "comparable" and the Δ can be computed by superimposing the two Petri Nets and highlighting the differences.

Detecting deviation: Differences between user task's Petri Net and user activity's Petri Net will be then automatically extracted at this stage. Indeed, activity Petri Net will indicate if there are abnormal activities (missing actions, new extra actions, abnormal sequence of actions) compared to the task Petri Net.

Mending task model: Once all deviations are detected and identified, it remains to update the user tasks and task model.

This process is iterative until validating Human Computer Interaction. It is also repetitive to ensure stability of the system when context change of the environment system occurs.

Our approach has two major purposes. First, it enables as to validate our interactive application design, especially user interfaces, through the identification of the mismatches between user tasks and user activities. Second, somehow deviations might inform us, when the design is already validated, about unexpected situation which pushes the user to act differently from what we have planned. For that matter, unexpected events could be identified by some criteria such as:

- User activity changes compared to what it used to be before (abnormal activity).
- User activity does not exist in history activity database.
- User activity does not exist in task model.

 ISBN: 978-1-61208-468-8

IV. CONCLUSION AND FUTURE WORK

This paper proposed a model-based approach for usability evaluation. The main feature of this approach is that evaluation is based on models. This approach is targeted to interactive applications that are already deployed and currently used.

Such an approach can increase the use of task models that are a very costly artifact, even for already deployed applications.

More precisely, we tried to log user activities considering elementary actions by using a correspondence controller. The latter will associate for each elementary action an elementary Petri Net [3]. This enables us to build the whole activity using a Petri Net in order to compare it to the user task modeled with the same language.

The main contribution of the paper lies in computing Δ between user tasks and user activities in order to detect all possible deviations or abnormal behaviors. That aims to mend and correct the task model.

We are currently applying this research in a case study of interactive application so as to expand more every step of our approach.

REFERENCES

[1] G. A. Boy, "From automation to tangible interactive objects", Annual Reviews in Control 38, April 2014, pp. 1-11, Elsevier

[2] M. Abed, H. Ezzedine, and C. Kolski, "Modelling tasks in design and evaluation of interactive systems : SADT/Petri method". In C. Kolski (Ed.), Analyse et Conception de l'IHM. Interaction Homme Machine pour les SI, vol. 1, 2001, pp. 145-174. Paris: Éditions Hermes

[3] I. Riahi and F. Moussa, "A formal approach for modeling context-aware Human-Computer System". Computers and Electrical Engineering, vol. 44, May 2015, pp. 241-261, doi:10.1016/j.compeleceng.2015.03.001

[4] J. Rassmussen, "Skills, rules and knowledge ; Signals, signs and symbols, and other distinctions in Human performance models", IEEE Transactions on systems, man and cybernetics, vol. smc 13, no. 3 May 1983, pp. 257-266

[5] F. Vanderhaegen, "Dissonance Engineering: A New Challenge to Analyse Risky", International Journal of Computers Communications and Control ISSN 1841-9836, December 2014, pp. 776-785

[6] C. Martinie, D. Navarre, P. Palanque, and C. Fayollas, "A Generic Tool-Supported Framework for Coupling Task Models and Interactive Applications", The seventh ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2015), June 2015, pp. 244-253, ISBN: 978-1-4503-3646-8

[7] P. Palanque, E. Barboni, C. Martinie, D. Navarre, and M. Winckler, "A Model-based Approach for Supporting Engineering Usability Evaluation of Interaction Techniques", ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2011), June 2011, pp. 21-30, ISBN: 978-1-4503-0670-6

[8] Q. Limbourg, C. Pribeanu , and J. Vanderdonckt, "Towards Uniformed Task Models in a Model-Based Approach". In: C. Johnson (ed.) DSV-IS 2001. LNCS, 2001, vol. 2220, pp. 164–182. Springer, Heidelberg

[9] R. Bastide and S. Basnyat, "Error Patterns: Systematic Investigation of Deviations in Task Models", Task Models and Diagrams for Users Interface Design, vol. 4385, October 2006, pp. 109-121, doi: 10.1007/978-3-540-70816-2_9

[10] F. Balagtas-Fernandez and H. Hussmann, "A Methodology and Framework to Simplify Usability Analysis of Mobile Applications", 24th IEEE/ACM International Conference on Automated Software Engineering, November 2009, pp. 520-524, doi: 10.1109/ASE.2009.12

[11] J. L. Crowley, P. Reignier, and R. Barranquand, "Situation Models: A Tool for Observing and Understanding Activity", Proceedings of the IEEE International Conference on Robotics and Automation ICRA 2009 Workshop on People Detection and Tracking, May 2009

[12] ISO 9241: Ergonomics Requirements for Office Work with Visual Display Terminals (VDTs) International Standards Organisation, Geneva (1997)

[13] J. Nielsen, Usability engineering, 1994, Morgan Kaufmann Pub.

[14] F. Paternò and C. Santoro, "Preventing user errors by systematic analysis of deviations from the system task model", Int. J. Human-Computer Studies, February 2002, vol. 56, issue 2, pp. 225–245 doi:10.1006/ijhc.2001.0523

 ISBN: 978-1-61208-468-8