Model-Based Usability Evaluation and Analysis of Interactive Techniques

Jean-François Ladry, Philippe Palanque, Eric Barboni, David Navarre

Model-Based Usability Evaluation and Analysis of Interactive Techniques

Jeff Ladry, Philippe Palanque, Eric Barboni & David Navarre
IRIT - University Paul Sabatier
118, route de Narbonne, 31062 Toulouse Cedex 9, France
{ladry, palanque, barboni, navarre}@irit.fr

ABSTRACT
This position paper presents a model based approach supporting development of advanced user interfaces for the design, simulation, tuning and the assessment of interaction techniques. It is based on a double concept: the introduction of additional information in models to allow designer to tune easily the interaction technique and the use of simulation and logging facilities to assess perform performance evaluation of the models. It proposes an alternative to user testing which is very difficult to setup and interpret when advanced interaction techniques are concerned.

Author Keywords
Model-Based approaches, formal description techniques, performance evaluation, multimodal interfaces, interactive software engineering, tuning.

INTRODUCTION
Using models for the design of computer systems provide a description of the system abstracted away from its implementation. Nowadays, such approaches are prominent in the area of software engineering via the Model Driven Engineering [9] field that emerged from the UML standard [11]. Indeed, as they provide a more abstract description of the system than the implementation code they also provide a unique opportunity for various stakeholders (users, developers…) to comment and propose modifications.

In The HCI community many researchers have described user interface elements by means of models. The interested reader can find a complete state of art of model-based user interface in [10] where the different modeling techniques are categorized by criteria such as: *Language*, *Interaction Coverage*, *Scalability, Tool support and Expressiveness.*

Beyond this descriptive aspect, models can also be used to support the evaluation of the user interface for properties (such as liveness and safety) or even for usability including: Model Based Usability remote evaluation as in RemUsine [13], EvaHelper [5], or in ReModEl [5]. Similar work can also be found for the Web domain as in AWUSA [14]. Usability evaluation can also be found for more generic purpose as in MeMo&MASP [6] or in [1] with MDA (Model-driven architecture)-compliant methods to improve software usability through model transformations.

Among these contributions, many have shown that HCI concerns must be integrated within the development process in order to design and develop usable systems. This is known as the "too little too late" problem detailed in [8] claiming that usability must be considered in the early stages of the development process or it will be only partially addressed.

Next section presents a model-based approach proposing an emphasis on *evolvability* and *modifiability* of models to be able to take into account usability concerns. The basic idea is to prepare models for modification at design time in order to be able to adjust them according to usability evaluation results.

THE APPROACH
This position paper proposes a design process for the design, simulation, tuning and assessment of interaction techniques.

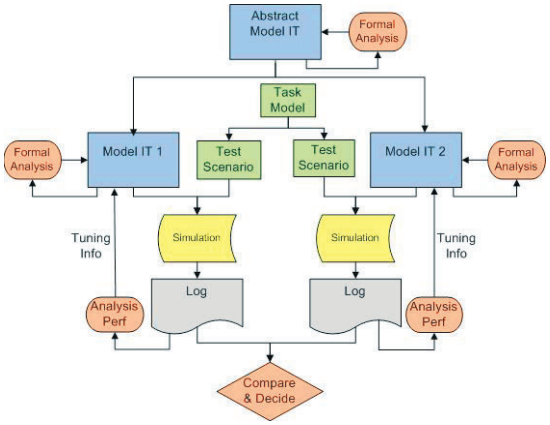

Figure 1 Process involving Interaction techniques, tasks models and Analysis +Log

Figure 1 presents the process of this approach exemplified for the comparison of two interaction techniques.

Modeling Interactions techniques
In the beginning of the process an abstract model (Abstract Model IT) is constructed using the ICO formalism [10] to accomplish the task represented in the Task model in CTT. From abstract model, multiple detailed models can be produced. These models refine the abstract model according

Pre-proceedings of the 5th International Workshop on Model Driven Development of Advanced User Interfaces (MDDAUI 2010): Bridging between User Experience and UI Engineering, organized at the 28th ACM Conference on Human Factors in Computing Systems (CHI 2010), Atlanta, Georgia, USA, April 10, 2010.

for instance to properties we want the technique to fulfill or according to the different modalities that have to be used. From the task model in CTT, a test scenario is extracted for each ICO model that the interaction technique must be able to perform. The detailed model is then simulated in Petshop according to the test scenario.

Simulation and Logging

During this simulation a log file is produced containing all the information about the evolution of the model.

	type	name	action	time	class	multiplicity	tokeninfo
581	place	currentxy	token_removed	1,25484E+12	Drag&Drop	1	<331,343>
582	place	currentxy	token_added	1,25484E+12	Drag&Drop	1	<549,361>
583	firetransition	NotOnTrash	none	1,25484E+12	Drag&Drop		
584	transition	OnTrash	substitution_changed	1,25484E+12	Drag&Drop		
585	transition	mouseMove_t1	substitution_changed	1,25484E+12	Drag&Drop		
586	transition	Trash	substitution_changed	1,25484E+12	Drag&Drop		
587	transition	NotOnTrash	disabled	1,25484E+12	Drag&Drop		
588	transition	mousePressed_t1	substitution_changed	1,25484E+12	Drag&Drop		
589	transition	mouseMove_t2	substitution_changed	1,25484E+12	Drag&Drop		
590	transition	Notrash	substitution_changed	1,25484E+12	Drag&Drop		

Figure 2 Excerpt of a Model-based log

The model-based log (presented in Figure 2) records all the change which occurs in the model during the simulation including firing of each transition, the removal and addition of a token in every place. This data is then exploited to assess the performance of each interaction technique in absolute value as well as their relative performance. If the performance does not fit the expectations, the log data can be used to modify or tune the model that will be simulated again. Such modification or tuning is made much easier as the information in the log is already related to the structure of the model.

Formal Analysis

Due to the Petri nets-based roots of the ICO formalism, we are able to use Petri nets properties analysis techniques such as place and transition invariant. These invariant allow us to prove properties such as liveness of a transition in a model for example. In the case of an interaction technique this would make it possible to assess that the transition handling mouse move events is always available (it is always possible to produce such events by moving the device.) According to the result of the analysis process, it can be decided to modify the model.

Tuning of models (Evolvability and Modifiability)

According to the performance evaluation obtained with the log analysis, some fine tuning can be applied in the model to increase the performance of the interaction technique. This fine tuning can either be done during or before the simulation as in PetShop models can be modified while executed.

Figure 3 presents the drag and drop interaction technique modeled using the ICO formalism. In the initial state, the interaction technique is Idle (there is a token in place *Idle*), the position x,y of the mouse cursor is stored as a token in place *Currentxy*, the reference to the graphical object frame (where the cursor is moving) is stored as a token in place *Frame* and the reference to the object trash is stored as a token in place *Trash*. From that initial state two transitions are available (represented in darker grey in the model: mouseMove_t1 and mousePressed_t1. Transition mouseMove-t1 is fired when the corresponding event is triggered by a user action on the input device.

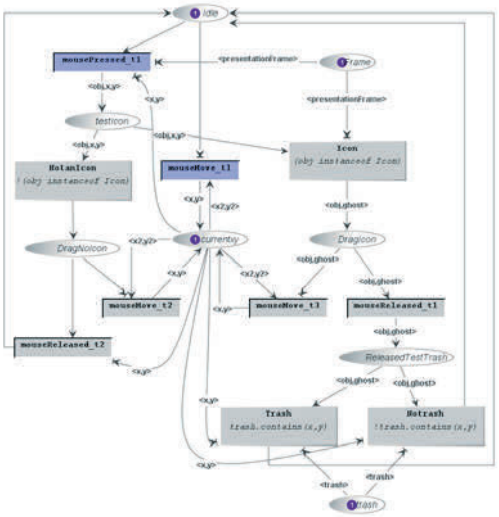

Figure 3 ICO model of basic Drag & Drop interaction technique

When this occurs, the value of the token stored in place *Currentxy* is changed to contain the new position of the cursor. Transition mousePressed_t1 is triggered by a user action on the button of the mouse. When this occurs, the model tests (represented by transitions NotonIcon and Icon) if the cursor is currently on an icon or not. If the cursor is on an icon then the model will process mouse move events (transition mouseMove_t3 which updates the cursor position) and mouse released events. When a mouse released event is received the model tests if the position of the cursor is on the icon of the trash (transition Trash) or not (transition Notrash). If yes, the file is deleted (this is modeled in the code of transition Trash and not represented here due to space constraints).

This model is not easily modified to integrate tunings that are currently made on drag and drop techniques such as acceleration and deceleration of the icon (according to the proximity of the target icon or according to the rapidity of the movement). However, it represents precisely and without any ambiguity the desired behavior of the initial interaction technique. According to our experience with interaction technique modeling, we know that fine tuning of the interaction technique is required.

Figure 4 presents and extended version of the model of Figure 3 adding possibility for tuning the interaction technique. Two new transitions have been added (in blue in Figure 4) allowing the possibility to check if the pointer is on the reactive object (here the trash) or not. With this information we can easily change the speed of the pointer when it is on the reactive object to “stick” it on the object for example. Such modification corresponds to changes in the motor space as introduced by [4].

To make it possible to tune this interaction, we have also added *Acceleration* and *Deceleration* places (circled in red in Figure 4) and linked them to MouseMove_t2 transition.

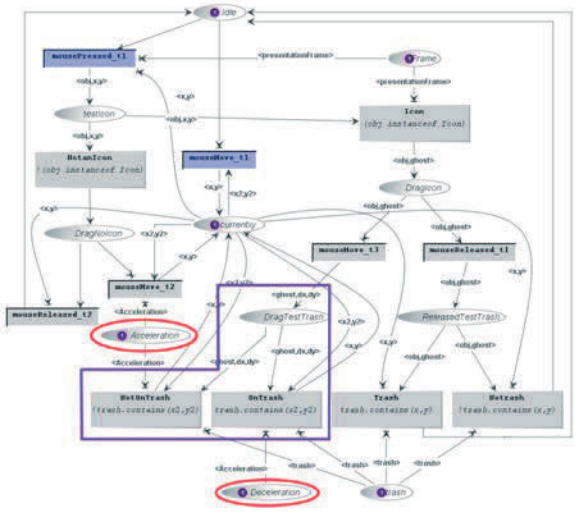

Figure 4 ICO model of tunable Drag & Drop interaction technique

The *Acceleration* place contained a value used for the acceleration of the mouse cursor when an object is dragged;

The principle of the approach is to run simulations of the models to identify possible limitations and propose modifications to be made in the models to improve the efficiency of the interaction technique. After tuning a new simulation is performed and the results are compared to the desired properties.

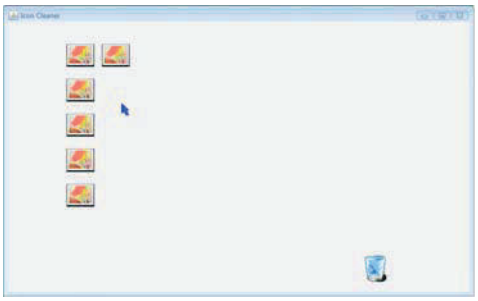

Figure 5 User Interface of the case study

Next section sketches how this approach can be instantiated on a case study, presenting the models, the transformations and the results of the performance evaluation.

CASE STUDY

The objective of the case study is to present the various phases of the approach on a simple but realistic application (see Figure 5). In the application a set of icons are presented in a window on a grid. The user's goal is to remove all the icons on the user interface by doing, iteratively in any order, the selection of an icon and the triggering of a deletion command the selection and deletion of icons. To support this goal two different interaction techniques have been modeled. Following the terminology of Figure 1, Model IT1 (called Drag & Drop) features an interaction technique of type Drag and Drop and behaved as described in the previous section. Model IT2 (called Speak & Click) features a multimodal interaction technique involving speech recognition (for the deletion command) and gesture (for icon selection). Systems and tasks models related to these two interaction techniques are presented in the next sections.

Modeling Interaction Technique 1

The behavior of IT 1 is presented in Figure 4. According to the more detailed description of the interaction technique, the abstract tasks to be refined as presented in Figure 6 in order to produced test scenarios (as presented in the design process of Figure 1). Selection is performed first by deciding the icon to be deleted then by moving the mouse cursor on the icon and by pressing the mouse button. Deletion is performed by moving the selected icon over the trash icon, verifying that trash icon is highlighted and releasing mouse button.

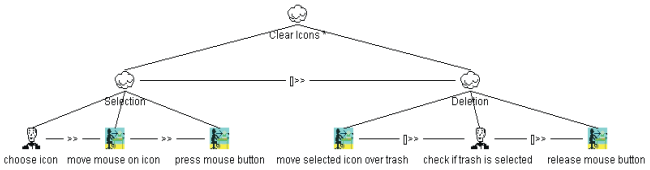

Figure 6 Task model refined to be conformant with Drag & Drop behavior

It is interesting to note that the temporal operator between tasks Deletion and Selection is *order independence*. The same imposed sequencing can be found in the ICO model where the model TI 1 (in Figure 4) imposes to start interaction with the selection (deletion is only available later on).

Modeling Interaction technique 2

Similarly to what has been done for Interaction technique 1, Interaction Technique 2 has been fully described using the ICO formalism and is presented on Figure 7.

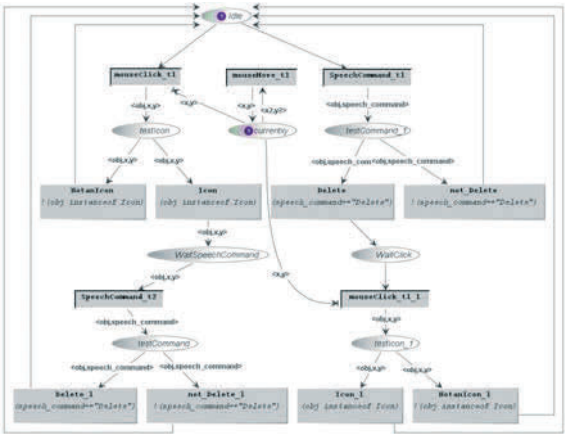

Figure 7 ICO model of system B (Speak & Click interaction technique)

This model allows users to either start by a speech command "*delete*" and then selecting an icon or selecting first an icon to be deleted and then uttering the word "*delete*". The abstract tasks model has to be refined similarly to the model in Figure 6.

Simulation

Simulation of the interaction technique models is done in the case tool Petshop. Further information about the case

tool can be found in [4] and about the simulation in. We don't provide here more information about the simulation as it has been introduced in [2] and is beyond the scope of this position paper.

Logging

From log extracted from the simulation of IT1 we can produce information such as the total time for a Drag&Drop. We can also compute the number of time the move change from *OnTrash* to *notOntrash* before the releasing to represents the number of time the user has missed the trash and exited the target without releasing on the icon. All such information comes only from the places and transitions that can thus easily be seen on the model represented in Figure 4 and the relationship with the log as presented in Figure 2 is immediate.

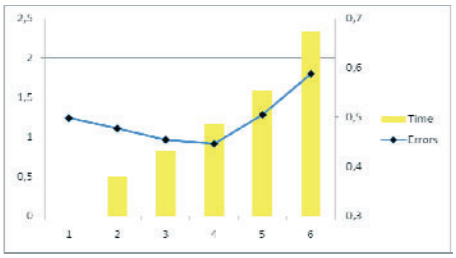

Figure 8 Result of Analysis of Log

After identifying where that information will be extracted from the model extract, we can simulate several time the model with different values and see if the total speed of the interaction technique and the number of errors to execute this task evolve. Such results can be gathered in a graph as presented in Figure 8. That graph shows that for an increase of acceleration of the mouse (Acceleration place in Figure 4), first the Drag&Drop is faster. But when the acceleration is 5, the errors are too important and the time to make the Drag&Drop increases and becomes worst than the standard interaction technique without acceleration

CONCLUSION

In this paper we have presented an approach to test and evaluate different interaction technique. This testing and evaluation is driven by models. With these models we can also tune finely different aspects of the interaction technique. This approach has exemplified on a small example where we show the interest of a model based usability evaluation. The results show well known results in HCI such as that acceleration improves efficiency of target selection to a certain extend. The objective of the approach is to apply it to novel and more sophisticated interaction techniques (possibly multimodal ones) which are much harder to assess especially through user testing.

REFERENCES

1. Abrahão S., Iborra E., and Vanderdonckt J. 2007. Usability Evaluation of User Interfaces Generated with a Model-Driven Architecture Tool. In Maturing Usability: Quality in Software, Interaction and Value, Series: Springer Human-Computer Interaction Series, Vol. 10, E. Law, E. Hvannberg, and G. Cockton (Eds.), 610p., 2007, ISSN: 978-1-84628-940-8, Springer.
2. Bastide, R., Navarre, D., and Palanque, P. 2002. A model-based tool for interactive prototyping of highly interactive applications. In CHI '02 Extended Abstracts on Human Factors in Computing Systems (Minneapolis, Minnesota, USA, April 20 - 25, 2002). CHI '02. ACM, New York, NY, pp. 516-517.
3. Balagtas-Fernandez F., Hußmann H., A Methodology and Framework to Simplify Usability Analysis of Mobile Applications In Proceedings of the 24th IEEE/ACM International Conference on Automated Software Engineering (ASE 2009). Auckland New Zealand, November 2009, ISBN 1527-1366/09, pp. 520-524.
4. Blanch R., Guiard Y. and Beaudouin-Lafon M. Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation. In Proceedings of CHI 2004, pages 519-526, Vienna - Austria, April 2004.
5. Buchholz G, Engel J, Märtin C, Propp S. Model-based usability evaluation - evaluation of tool support. HCII 2007, Beijing, China, 2007. p. 1043-52. Springer-Verlag, Berlin, Germany, 0302-9743
6. Feuerstack, S., Blumendorf, M., Kern, M., Kruppa, M., Quade, M., Runge, M., and Albayrak, S. 2008. Automated Usability Evaluation during Model-Based Interactive System Development. In Proceedings of the 2nd Conference on Human-Centered Software Engineering and 7th international Workshop on Task Models and Diagrams (Pisa, Italy, September 25 - 26, 2008). P. Forbrig and F. Paternò, Eds. Lecture Notes In Computer Science, vol. 5247. Springer-Verlag, Berlin, Heidelberg, 134-141.
7. Kristoffersen, S. 2009. A Preliminary Experiment of Checking Usability Principles with Formal Methods. In Proceedings of the 2009 Second international Conferences on Advances in Computer-Human interactions (February 01 - 07, 2009). ACHI. IEEE Computer Society, Washington, DC, 261-270
8. Lim, K. Y. and Long, J. (1994). The Muse Method for Usability Engineering. Cambridge University Press.
9. Mellor S. and Balcer M. (2002). Executable UML: A Foundation for Model-Driven Architectures. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.
10. Navarre, D., Palanque, P., Ladry, J., and Barboni, E. 2009. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM Trans. Comput.-Hum. Interact. 16, 4 (Nov. 2009), 1-56.
11. Object Management Group (2003). Unified Modelling Language (UML) 2.0 Superstructure Specification, August 2003. Ptc/03-08-02, pp. 455-510
12. Paternò F., Mancini C., Meniconi S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models, Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction, p.362-369, July 14-18, 1997
13. Paternò, F., Russino, A., Santoro, C. (2007) Remote evaluation of Mobile Applications, Task Models and Diagrams for User Interface Design 6th International Workshop, TAMODIA 2007, Toulouse, France, November 7-9, 2007, Lecture Notes in Computer Science , Vol. 4849, Winckler, Marco; Johnson, Hilary; Palanque, Philippe (Eds.) ISBN: 978-3-540-77221-7
14. T. Tiedtke, C. Märtin, N. Gerth. Awusa, A tool for automated website usability analysis. 9th Int. Workshop DSVIS, 2002.