Model-driven Engineering for Optimizing the Usability of User Interfaces

Marwa Hentati[1], Lassaad Ben Ammar[1], Abdelwaheb Trabelsi[2] and Adel Mahfoudhi[3]
[1]*University of Sfax, National School of Engineering, CES Laboratory, Sfax, Tunisia*
[2]*University of Sfax, Sfax, Tunisia*
[3]*College of Computer and Information Technology, Taif University, Taif, Saudi Arabia*

Keywords: MDE, User Interface, Usability Optimization, Transformation Process.

Abstract: Usability is considered to be one of the most important quality factors that determine the success or failure of an interactive system. This can be explained by the ever-increasing number of studies addressing the integration of the usability evaluation at the development process. However, most of these proposals aim to guide the user interface transformation process according to a set of usability criteria allowing the generation of user interfaces which meet the usability requirement. In fact, the selection of the adequate alternative transformation depends on the usability attributes that will be met. This paper proposed an approach that considers the user interface generation process as a usability optimization problem according to given usability optimization objectives. The aims to generate all possible concrete user interfaces from a given abstract user interface. Then, the usability optimization process selects the optimal concrete user interface for a specific context of use.

1 INTRODUCTION

The model-driven engineering (MDE) is currently being adopted in the Human-computer Interaction (HCI) field to support user interface specification and engineering activities (Hussmann et al., 2011). The MDE paradigm is proved to be quite appropriate (Favre, 2004). This approach specifies an automated process of developing user interfaces through the definition of models and their transformation from high-level models to code generation in the target platform (OMG, 2003). A renowned approach in this context is the Cameleon Reference framework (Calvary et al., 2003). It provides a unifying reference framework that structures the user interface development taking the context of use (user, platform and environment) into account.

In this framework, focus is generally placed on data and functional modelling, disregarding usability aspects. Therefore, there is a need to extend the MDE process in order to promote usability as a first class entity in the development process.

The present paper aims to delineate an approach that addresses these weaknesses by extending the user interface process in order to optimize the user interface usability. The proposed approach is intended to optimize the usability from the conceptual model in a user interface (UI). It should be noted that the conceptual model covers the abstract user interface (AUI) model and the concrete user interface (CUI) models. Therefore, the selection of the adequate alternative transformation depends on the usability attributes, which are able to convey. The proposition is structured in three main contributions: (1) generating all the possible concrete user interfaces from a given abstract user interface, (2) formulating the usability optimization function for a given context of use and (3) selecting the alternative transformation able to generate the optimal usability user interface.

We structure the remainder of this paper as follows: section 2 presents some related studies. Section 3 describes the different stages of the proposed approach. In order to show the usefulness of our proposal, a case study is presented in Section 4. Finally, section 5 presents the conclusion and the future research work.

2 RELATED WORKS

There are currently several research studies that have dealt with usability in MDE environment have been proposed. The main objective of these methods is to propose a set of usability attributes in order to drive the selection of adequate alternative transformations.

In (Panach et al., 2013), the authors have addressed the usability features related to the system functionality, which may involve important changes in the system architecture. They use the term FUFs (Functional Usability Features) to indicate this type of usability features. The FUFs are abstractly represented by means of conceptual primitives that will extend the conceptual model of an existing model-driven development (MDD) method. Then, the conceptual model can be seen as the input of a model compiler that can generate the software application automatically (or semi-automatically). In (Panach et al., 2014) a set of the well-known FUFs in the human–computer interaction (HCI) community are gathered in a usability model that is included from an early stage of a holistic MDD method.

In (Huerta et al., 2010), the authors have proposed an architecture to perform the quality of the model-driven transformation. The main goal of this architecture is to define a set of artifacts and a process for specifying and executing model transformations. The selection of the alternative transformations is guided by quality attributes. The feasibility of the proposed architecture is shown using a case study which transforms a requirements model into a UML class model.

(Ammar et al., 2014) have suggested an approach to integrate usability issues as a part of a user interface development process from the conceptual models. The proposed approach is structured in two main stages: (1) the definition of the model transformation and (2) the execution of a parameterized transformation. Concerning the first stage, it establishes a set of transformation rules describing all the possible alternative transformations for a given domain. As for the second one, it executes a model driven transformation parameterized by the usability model in order to select the adequate alternative transformation.

Although the previously mentioned works are useful in the interplay between the usability and MDD paradigm, they lack precise details about the quality of the alternative selection process. In fact, all of them try to select the alternative transformation without addressing the problem of the choice optimization. This problem is the object of some initiatives in the research field among which we quote (Gajos et al., 2010), (Petter et al., 2008) and (Raneburger et al., 2011).

In (Gajos et al., 2010), the authors implemented the SUPPLE system that automatically generates multi-target UI using the user interface specification as input. The user interface adaption is treated as an optimization problem related to a user and device specific cost function. Therefore, the SUPPLE system renders each interface element from the functional interface specification into an appropriate concrete widget according to a matching function. To find the optimal one, SUPPLE relies on a cost function that provides a quantitative metric, such as the speed of use.

In the same context, (Petter et al., 2008) proposed to optimize the usability of the graphical user interface (GUI). They suggest an extension to the QVT standard in order to consider the usability aspects. The main goal is to transform each element from the user interface specification model to an optimal (GUI) component. The optimization function has been formulated on the basis of the manipulation and navigation time.

Face to the increasing use of small screen devices, (Raneburger et al., 2011) suggested an approach for extending and incrementing the Discourse-based communication models transformation process. This approach introduces straight-forward optimization techniques into the model-driven generation of GUIs to reduce some usability problems. This allows the optimization of the generated GUIs for devices with small screens in such a way as to utilize the given space and to minimize navigation and scrolling.

The analysis of the aforementioned studies allows us to detect some limitations related to the usability optimization problem. In fact, a limited usability aspect may not ensure the usability of the final application. It is within this context that our research work lies to introduce a framework addressing the problem of usability optimization in an MDE-compliant method.

3 USER INTERFACE GENERATION AS USABILITY OPTIMIZATION PROBLEM

3.1 Overview

Since its first beginning HCI has been concerned with the evaluation of interactive systems, and consideration of usability. Recently, the MDE has been adopted in the HCI domain to support interaction systems specification and engineering activities (Hussmann et al. 2009). In this context, the approach is completely based on the Cameleon framework which presents a unifying framework for the development of multi-target user interfaces supporting multi-context of use (Calvary et al., 2003). In the development process, the unifying framework structures the user interface development process into four levels of abstraction, starting from task specification to a running interface: The Task and Concepts (T&C) level, the Abstract User Interface (AUI) level, the Concrete User Interface (CUI) level and the Final User Interface (FUI) level. It should be noted that the proposed approach is intended to improve the usability at the conceptual model which covers the AUI model and the CUI model. In fact, the conceptual model represents an abstraction of the user interface. The proposed approach is made up of three main stages:

- Generating all the possible CUI from a given AUI,
- Optimizing the usability at the conceptual model,
- Selecting the alternative transformation able to generate a user interface with an optimal usability.

In the first stage, all the possible generation of CUI are established from a given AUI. The second stage performs a usability optimization process taking as input each CUI established in stage one. A profit usability function is formulated as an optimization function according to a set of usability attributes. With each one of them is associated at least one metric which provides a numerical value interpreted by a mechanism of indicators. In the third stage, the user interface with the optimal value of usability will be selected as a result of the optimization process in a particular context of use. Figure 1 shows an overview of the proposed approach.

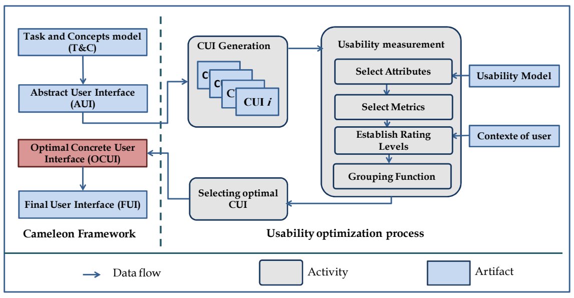


Figure 1: The proposed approach to optimize the usability of UI with model-driven development process.

Each stage of this approach will be described in the following sections.

3.2 The Concrete User Interface Generation

Model transformation is a key mechanism when building software systems with MDE approach. The model transformation expresses how one or more source models are transformed into one or more target models. It consists of a set of transformation rules, each of which describes how a construct of the source model can be transformed into another in the target model. The elaboration of the transformation rules identifies the corresponding construct in the CUI model for each construct from the AUI model. There may be more than one construct in the CUI model that can be associated with the one from the AUI model.

Transformation associated with an abstract component are equivalent. However, they may differ from the non-functional perspective and do not satisfy the same usability attributes. For example, in Figure 2, the first solution displays the ranges of the accepted values. Hence, the user is well guided to insert the expected doors number, and therefore, the Prompting attribute is well fulfilled. In the second and third solutions, the user is generally intended to select from a set of possibilities, and thus the Error Prevention attribute is well fulfilled. By considering this example, we can notice that each transformation optimizes at least one usability property.

In general, we can calculate the number of CUIs (CUI_{num}) that can be generated for a given AUI as:

$$CUI_{num}=\prod_{i=1}^{n} transformation\_number_i \qquad (1)$$

where n is the number of abstract components that are matched in the AUI and the *transformation_number* is the number of all possible transformations associated with an abstract component i.

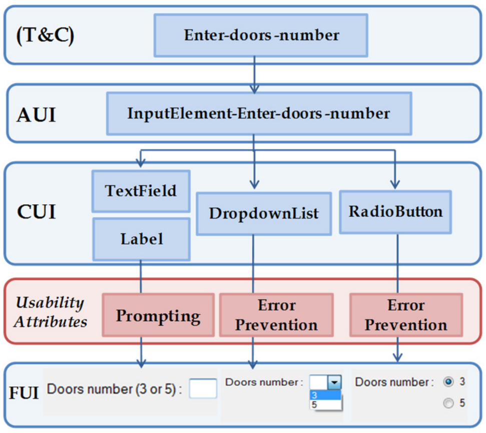


Figure 2: Example of different model transformations for the input data element.

3.3 Usability Measurement

Usability is a difficult concept to quantify. It involves several dimensions, mainly the process and the product dimensions. In this paper, we will focus on the product dimension since our objective is to evaluate usability at an early stage. Therefore, we opt to use a usability model which extends the one presented in the ISO/IEC 9126-1 standard (ISO, 2001). This view is more deal with the characteristics of the product itself and can be used to evaluate the intermediate artifacts produced in an MDE development process.

A usability optimization approach is implemented to optimize the usability of the targeted user interfaces against the defined usability attributes by means of usability metrics carried out at the conceptual model. To do so, we propose a usability optimization function that was formulated according to a set of usability attributes through the usability metrics. Consequently, this stage involves four main steps: (1) formulating a profit function, (2) selecting metrics, (3) establishing rating levels, and (4) grouping function. The last three steps are adapted from the evaluation method proposed by (Ammar et al., 2015).

3.3.1 Formulation of the Profit Function

This step aims to formulate a profit function which plays the role of an optimization objective function. The profit function is intended to select the optimal CUI generated in the first stage (section 3.2) of the proposed approach.

Our optimization objectives reflect which aspect to be considered in our approach and in which order we consider their importance. In this context, we believe that the usability optimization can be improved by means of optimizing the *performance* and the *appearance* of the user interface. Hence, we formulated our usability optimization problem through two optimization objectives which are:

- Maximizing the user performance P( ),
- Maximizing the user interface appearance A().

We start by defining the ***CUIprofit*** of a specific combination of transformation (t_i) to be of the form:

$$\textbf{\textit{CUIprofit}}\ (t_i) = P(t_i) + A(t_i) \qquad (2)$$

where $P(t_i)$ represents the performance value of a user interface generated with a specific transformation t_i and $A(t_i)$ represents the appearance value of a user interface generated with a specific transformation t_i.

Since our objective is to optimize the usability before the implementation of the user interface, we believe that the user performance is optimized when the user is well informed, guided and prevented from making mistakes during the interaction with the final user interface. In addition, optimizing the user performance needs to minimize the number of steps required to accomplish a task and an accepted level of the control action which includes cancellability, undoability and explicit user actions. Indeed, we define that the user ***performance*** value of a given CUI which represents objective 1 as the following formula:

$$P(t_i) = W_{BR} * BR(t_i) + W_{ERP} * ERP(t_i) + W_{PR} * PR(t_i) + W_{UCA} * UCA(t_i) \qquad (3)$$

where $BR(t_i)$ represents the brevity value, $ERP(t_i)$ is the error prevention value, $PR(t_i)$ is the prompting value, $UCA(t_i)$ is the user control action value and W_{BR}, W_{ERP}, W_{PR}, W_{ERP} represent the additional weight values.

While the user performance is of high importance, the user interface appearance is an equally essential factor for determining the user interface usability that gives the user a comfortable feeling during the interaction with the user interface (Marcus, 2011). Therefore, the usability is enhanced by improving the appearance of the user interface (or synonyms and related concepts, such as "aesthetics", "attractiveness", "beauty" or "form"). (Ngo-2000).

We believe that obtaining an optimized user interface appearance needs an appropriate number of elements per window by keeping a good equilibrium between information density and white space in a specific computing-device. Additionally, the best appearance can be ensured by the consistency of the user interface elements and the presence of an acceptable number of navigation elements. Thus, the ***appearance*** A () value of a specific CUI which represents objective 2 is calculated by the following formula:

$$A(ti) = W_{ID} * ID(t_i) + W_{CN} * CN(t_i) + W_{NAv} * NAV(t_i) \qquad (4)$$

where $ID(t_i)$ represents the information density value, $CN(t_i)$ is the consistency value, $NAV(t_i)$ is the navigability value and W_{ID} W_{CN} W_{NAv} represent the additional weight values.

It should be noted that each additional weight (i.e., W_{BR}, W_{ERP}, W_{PR}, W_{UCA}, W_{ID}, W_{CN}, W_{NAv}) is used to influence the impact of a specific addend on the overall usability profit value. The importance of the usability attributes varies according to systems and target population characteristics (e.g., small screen device, user experience). This work was described with details in (Hentati et al., 2015).

3.3.2 Selecting Metrics

We need to associate with each usability attribute at least one metric. In what follows, we illustrate for each usability attribute the opted metric and its generic description:

Brevity (BR): The user interface should allow users to interact with the UI element in a few number of action steps. The brevity can be quantified by the number of steps (counted in keystrokes) required to accomplish a task.

$$BR = \textit{distance}(x,y) \qquad (5)$$

where *distance(x,y)* represents the distance between source screen (x) and the target screen (y).

Error Prevention (ERP): can be quantified by the percentage of the list primitive used instead of text field when the input element has a set of enumerated values:

$$ERP = \text{list}(x)/n \qquad (6)$$

where list(x) represents the number of the list primitive and *n* is the number of input data elements with limited possible values.

Prompting (PR): the prompting can be measured in terms of the proportion of label that displays supplementary information.

$$PR = \textit{StaticField()}/n \qquad (7)$$

where *StaticField()* represents the number of labels which display the supplementary information and *n* is the total number of static field (label).

User Control Actions (UCA): We propose to quantify the user control actions according to the degree of control assigned by the system which includes cancellability, undoability and explicit user actions.

$$UCA = \Sigma x_i / n \qquad (8)$$

where x_i represents the action elements and *n* represents the total number of elements.

Information Density (ID): The information density of a user interface can be measured in terms of the number of elements per user interface.

$$ID = n \qquad (9)$$

where *n* represents the total number of UI elements per interface.

Consistency (CN): This factor includes the element consistency attribute that can be measured by means of the percentage of the repeated elements and the label consistency which can be quantified by the means of the percentage of the repeated label at the same user interface.

$$CN = r \qquad (10)$$

where *r* represents the proportion of repeated elements with the same label.

Navigability (NV): The navigability measures the level of facilities that the system will provide to navigate throughout several interfaces. We propose the average of navigation elements per UI as an internal metric to measure the navigability attribute.

$$NV = n \qquad (11)$$

where *n* represents the number of navigation elements.

3.3.3 Establishing Rating Levels

The previously defined metrics provide a numerical value that necessitates having a meaning in order to be interpreted. The mechanism of indicator is restored in order to reach such goal. It consists in the attribution of qualitative values to each numerical one. In (Ammar and Mahfoudhi, 2013), the qualitative values was summarized in: very good (VG), good (G), medium (M), bad (B), and very bad (VB). According to (Panach et al., 2014), these indicators do not differentiate correctly between the values (VG) and (G), and the values B and VB. In our approach, we propose to attribute three indicators (B), (M) and (G). For each qualitative value, a numerical range was assigned. For example, the prompting value (PR) may be considered as good (G) when at least 95% of input element labels should display additional information (e.g. the required data format). Table 1 shows the list of indicators that have been defined.

Table 1: Proposed indicators for usability metrics.

Metric	G	M	B
PR	$PR \geq 0.95$	$0.75 \leq PR < 0.95$	$PR < 0.75$
BR	$BR \leq 2$	$2 < BR \leq 5$	$BR > 5$
ERP	$ERP \geq 0.9$	$0.7 \leq ERP < 0.9$	$ERP < 0.7$
UCA	$UCA \geq 0.9$	$0.5 \leq UCA < 0.9$	$UCA < 0.5$
ID	$ID < 15$	$15 \leq ID < 25$	$ID \geq 25$
CN	$CN \geq 0.85$	$0.6 \leq CN < 0.85$	$CN < 0.6$
NV	$NV < 7$	$7 \leq NV < 12$	$NV \geq 12$

3.3.4 Grouping Function

The aim of this stage is to put metrics and attributes together in order to obtain a single usability profit measure. While executing the usability evaluation, the previously defined metrics are applied. The obtained numerical values are converted into their corresponding qualitative ones. Next, each categorical value is converted to numerical values with respect to the following hypothesis. (G) ⇒3, (M) ⇒2, (B) ⇒1. The final result is calculated by a sum function.

The last step in the grouping function is to convert the obtained numerical value into an ordinal value. We assign B to the value between 1 and 2, M to the value between 2.1 and G to value lower or equal to 3. The three steps are applied for each grouping. Groupings are performed bottom-up from indicators until the overall user interface usability is reached.

3.4 Selecting the Optimal CUI

The entire CUI generated in stage one has a final usability evaluation value. The higher one is selected as the optimal CUI. It should be noted that the present stage is actually performed (semi)-automatically. The selecting stage will be an exhaustive task with the increasing number of CUI possible combinations. That is why we hope to automate this stage by implementing it in the future.

4 AN ILLUSTRATIVE CASE STUDY

This section investigates a case study in order to illustrate the feasibility of our proposal. The research question addressed by this case study is: *Does the proposal ensure the usability optimization of the generated user interface artifact*?

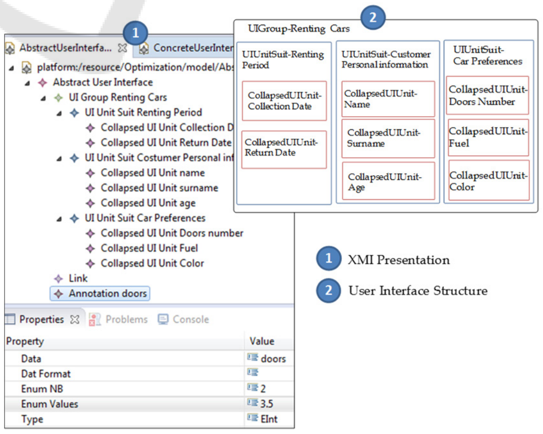
The object of the experimentation is Rent-a-car system. The scenario is adapted from (Bouchelligua et al., 2010). The system will run on terminals of customers (laptop, PDA, mobile phone, etc.). Therefore, the user interface must be able to bring not only a feeling of comfort when interacting with a best appearance user interface but also a best performance regardless the context of use.

Since the Rent-a-car system was large, we focused our interests on the generation of the CUI for "the renting period", "customer personal information" and "car preferences" tasks. The transformation of the AUI allowed the generation of 216 CUIs. It covered all the possible combinations between the possible concrete components. It should be noted that the number of all possible combinations was obtained from a limited number of tasks (3 tasks and 8 sub-tasks), thus selecting the best alternative among them is an exhaustive task which can be solved by our approach according to the usability profit function.

We suppose to have the abstract user interface from Figure 3 as a result of the transformation of the three tasks «the renting period", «customer personal information» and «car preferences» following the model transformation explained in details in (Bouchelligua et al.,2010). The result of the transformation is an XML file which is in accordance with the AUI metamodel (left part of Figure 3). An editor with the Graphical Modelling Framework (GMF) of eclipse was developed in order to better clear up the user interface layout. The sketch of the user interface presented by the editor is shown in the right part of Figure 3.

The AUI contains a *UIGroup* called «Renting Cars» which gives access to three *UIUnitSuit* called «Renting Period», «Customer Personal information» and «Car Preferences». The «Renting Period» container gives access to the collection date and the return date. The customer should specify his/her name, surname and age whose acceptable value should be more than 18. After that, the customer should specify the car preferences as the doors number, fuel and color.

To better explain our proposal, we started the case study by a CUI model transformation illustrated in Figure 4. The generated CUI did not take into account our proposed usability optimization in a specific context of use. The context is the following: a laptop as an interactive device (normal screen size), a customer with a low level of experience of using the Rent-a-car system. Although the deducted CUI filled their functional objectives, it did not satisfy our proposed usability optimization profit functions.



```
platform:/resource/Optimization/model/Abs
  Abstract User Interface
    UI Group Renting Cars
      UI Unit Suit Renting Period
        Collapsed UI Unit Collection D
        Collapsed UI Unit Return Date
      UI Unit Suit Costumer Personal inf
        Collapsed UI Unit name
        Collapsed UI Unit surname
        Collapsed UI Unit age
      UI Unit Suit Car Preferences
        Collapsed UI Unit Doors number
        Collapsed UI Unit Fuel
        Collapsed UI Unit Color
    Link
    Annotation doors
```

Property	Value
Data	doors
Dat Format	
Enum NB	2
Enum Values	3.5
Type	EInt

Figure 3: The abstract user interface model for Renting-a-car system.

Each input element generated CUI illustrated in Figure 4 was transformed into *UIFieldEdit*. Since novice users should be protected against errors while entering data, input elements with limited values (eg. Doors number, Fuel and Car color) should be displayed with list box elements such as dropdown list and radio button. Moreover, novice users need to be well informed and guided by the presence of supplementary information displayed in the labels *UI StaticField* in order to ensure the best user performance.

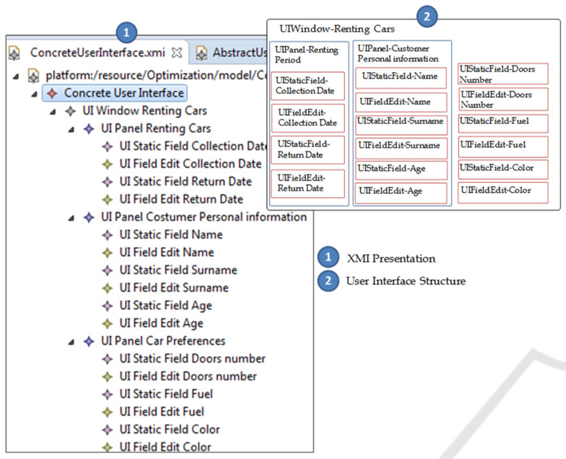


Figure 4: Non-optimal concrete user interface for a normal screen size.

The execution of a proposed approach, wherein the usability optimization function is conveyed lead to the generation of the user interface which contains concrete elements that fulfil the desired usability attributes. Figure 5 illustrates such transformation taking into account the computing device and the target population characteristics. In this example, the target population is a customer with a low level of experience of using the Rent-a-car system a laptop as an interactive device (normal screen size) (15''). In this case, the optimized CUI illustrated in Figure 5 ensure the best performance by using list box elements and by the presence of the supplementary

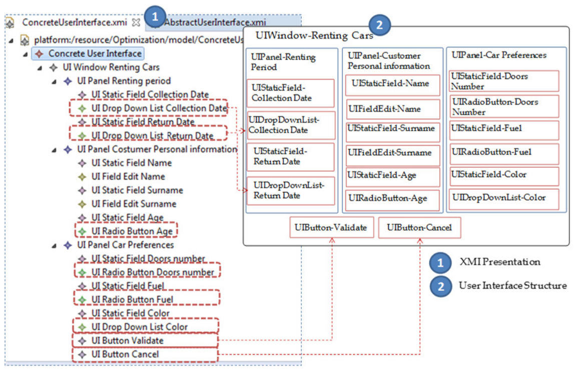


Figure 5: An optimal concrete user interface for normal screen size.

information like the mask of date (DD/MM/YYYY), the range of accepted values for the age value (>18 years) and the doors number (3 or 5). Hence, the novice users were well informed to insert the correct value. Furthermore, the UCA attribute is respected by the presence of a Cancel and Validate buttons.

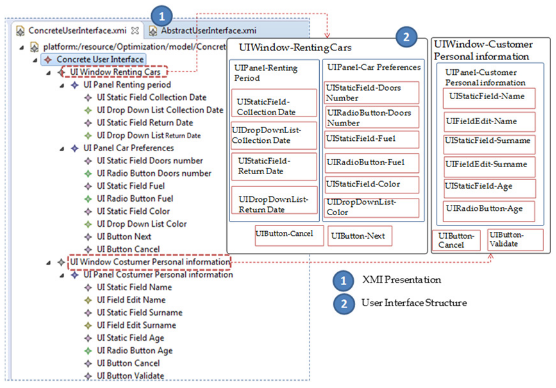


Figure 6: An optimal concrete user interface for a small screen size.

The second transformation to be conducted takes an «iPAQ Hx 2490 Pocket PC» as platform and a customer with a low level of experience of using the Rent-a-car system as a target population. The migration to such platform raises a new redistribution of the user interface elements. The small screen size (240x320) is not sufficient to display all information.

In Figure 6, the CUI with an optimal usability for small screen group a limited number of concrete components, whose maximum number of concepts can be manipulated (5 in the case of «iPAQHx2490 Pocket PC»). To ensure the best user interface appearance, the user interface elements are redistributed on several windows. The redistribution of interface elements on several windows will bring more steps to reach the goal by the aims of navigability elements (eg. Next, Return).

5 CONCLUSIONS

This paper presents an approach for optimizing usability aspects as a part of a user interface development process. It can be used in any software development proposal based on conceptual models. The proposed approach extends the Cameleon reference framework by integrating the usability optimization process from the conceptual models. This proposition gathers three main stages: the first one aims to generate all possible concrete user interfaces from a given abstract user interface. Then, a usability optimization process was performed. We used a profit function by a set of usability attributes, each of which was associated at least with a metric. These metrics will be interpreted by defining a mechanism of indicator taking into account the target population and the computing device characteristics. Finally, the usability results allowed the selection of the optimal alternative transformation.

Several research studies can be considered as a continuation of this work. In fact, the present approach can be defined as a linear optimization, while respecting a set of optimization constants. An empirical validation of the optimized user interface is recommended to clearly demonstrate the coherence between values obtained by our proposal and those perceived by end-user.

REFERENCES

Aquino, N., Vanderdonckt, J., Fernandez, N.C., Dieste, O., Pastor O., (2010) " Usability evaluation of multi-device/platform user interfaces generated by model-driven engineering", In: *Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement ESEM* '10, pp. 30: 1-30. DOI 10.1145/1852786.1852826.

Ammar, L. B., Trabelsi, A. and Mahfoudhi, A.,(2014) "Incorporating usability requirements into model transformation Technologies", *Requirements Engineering: 1-15, 2014.*

Ammar, L.B., & Mahfoudhi, A. (2013). Early usability evaluation in model driven framework. In *ICEIS 2013: Proceedings of the 15th international conference on enterprise information systems, Volume 3* (pp. 23–30). France: Angers, 4–7.

Ammar, L. B., Trabelsi, A., & Mahfoudhi, A. (2015). A model-driven approach for usability engineering of interactive systems. *Software Quality Journal,* 1-35.

Bouchelligua,W., Mahfoudhi,A., Mezhoudi,N., Dâassi, O., and Abed, M. (2010). User interfaces modelling of workflow information systems. In *EOMAS*, pages 143–163.

Calvary, G., Coutaz, J., Thevenin, D. (2003) "A unifying reference framework for multi-target user interfaces", *Interacting with Computers, vol.15,* no 3, p. 289-308.

Favre,J.M. (2004). Toward a Basic Theory to Model Driven Engineering.

Gajos, K. Z., Weld, D. S., and Wobbrock, J. O. (2010) "Automatically generating personalized user interfaces with Supple. Artificial Intelligence", 174(12), 910-950.

Hentati, M., Trabelsi, A., Ben Ammar, L., and Mahfoudhi, A. (2015). Towards optimizing the usability of user interface generated with model-driven development process. In *Human System Interactions (HSI), 2015 8th International Conference on (pp. 206-212). IEEE.*

Huerta, J.G., Blanes, D., Insfran E., and Abrahão, S.,(2010) "Towards an architecture for ensuring product quality in model-driven software development ", Proceedings of the 11th international conference on product focused software (PROFES '10). *ACM,New York*, NY, USA, pp 28–31.

Hussmann, H., Meixner, G., & Zuehlke, D. (Eds.). (2011). Model-driven development of advanced user interfaces, studies in *computational intelligence (Vol. 340). Berlin: Springer.*

ISO/IEC: ISO/IEC 9126 (2001). Software engineering: Product quality. *ISO/IEC.*

Marcus, A. (2011). Design, User Experience, and Usability. Theory, Methods, Tools and Practice: First International Conference, DUXU 2011, Held as Part of HCI International 2011, Orlando, FL, USA, July 9-14, 2011, *Proceedings (Vol. 6770). Springer.*

Ngo, D. C. L., Samsudin, A., & Abdullah, R. (2000). Aesthetic measures for assessing graphic screens. J. *Inf. Sci. Eng*, 16(1), 97-116.

OMG (2003) MDA guide version 1.0.1. http://www.omg.org.

Panach J.I., Aquino N. and Pastor O.,(2014) "A proposal for modelling usability in a holistic mdd method" *Sci. Comput. Program. 86*, pp 74-88.

Panach, J.I., Juzgado, N.J.and Pastor, O., (2013) "Including functional usability features in a model-driven development metho", *Comput Sci Inf Syst,10(3)*:999–1024.

Petter, A., Behring, A., and Zlatkov, M.,(2008) "Modeling usability in model transformations", In : *Proceedings of the 1st International Workshop on Non-functional System Properties in Domain Specific Modeling Languages, NFPinDSML*. p. 1613-0073.

Raneburger, D., Popp, R., Kavaldjian, S., Kaindl, H., and Falb, J., (2011) "Optimized GUI generation for small screens",Model-Driven Development of Advanced User Interfaces, volume 340 of Studies in *Computational Intelligence, Springer Berlin Heidelberg*, pp. 107-122.