# Filling the User Skill Gap Using HCI Techniques to Implement Experimental Protocol on Driving Simulators

Ghasan BHATTI[1,2,3],Guillaume MILLET[1]
*(1) OKTAL SA*
*Meudon, France*
*firstname.secondname@oktal.fr*

Roland BREMOND[2], Fabrice VIENNE[2],
Nguyen-Thong DANG[2]
*(2) Paris-Est University*
*IFSTTAR, LEPSiS*
*Paris, France*
*firstname.secondname@ifsttar.fr*

Jean-Pierre JESSEL[3]
*(3) IRIT, VORTEX Group*
*Paul Sabatier University*
*Toulouse, France*
*firstname.secondname@irit.fr*

***Abstract*—Programming activities are performed not only by programmers but also by end-users in order to support their primary goals in different domains and applications. End-users do not have formal training in programming, so interaction environment and systems are needed, which could account for user skills. The objective of our work is to fill the gap between the user skills and the goals they want to achieve using driving simulators. This paper presents the results of a research in which we have proposed a solution for the primary users of the driving simulator to design and implement experimental protocol. We have used the user-centered design (UCD) technique, conducted a user survey, and proposed a solution, in which we have categorized the Interface of the driving simulator into three sub-interfaces based on the skills of the users. These interfaces are Experiment Builder (Non-technical persons), Template builder (for technical persons) and Experiment Interface (for any user to run as experiment). A prototype based on this concept is developed and some feedback were collected from end-users. Our results indicate that, users can implement an experimental protocol without having programming skills using our proposed design.**

***Keywords*-Experimental protocol; User-Centered Design; HCI Techniques; Scenario modeling; Driving simulators**

## I. INTRODUCTION

Nowadays, programs are not only written by software engineers, but also by the people (end-users) who are not expert in programming, but they do programming in order to support their primary goals. e.g. scientist write program for analysis, teachers and accountant use spreadsheets for their support, and children can use them for games and simulation. What makes these end-user programmers different from professional software developers are their goals [1].
End-users usually do not have the time to be proficient in programming so they need processes, methods and tools with immediate feedback and results, which could account for the competencies they lack to achieve their goals using programming. One of the reasons why it is difficult, that programs are abstracts [2]. It is difficult for the end-users without a programming background to think in an abstract manner to implement the specific situation compared to thinking about the same situation in the real world. So abstraction is one of the barriers, especially for novice users. It is a common observation that, the more complex the situation, the more abstraction is required in the program. Also, programming languages have not been designed by addressing the human-computer interaction issues [3]. According to John et al. [4], user interface is one of the factors besides many other factors, because it provides an interaction between the end-user and computer. So there is a need to develop a user-centered interactive environment, which could support end-users to achieve their goals.

Driving Simulator is a useful research tool for behavioral researchers to study drivers behaviors, to analyze road safety features, and for drivers training without any safety risk. They are also used to evaluate ADAS (Advance Driving Assistance Systems). ADAS are the systems to help drivers during the driving process. They are designed with the purpose of increasing the road safety. The examples of these systems include ACC (Adaptive Cruise Control), Lane Change Warning (LCW), Automatic parking etc. In order to retrieve the data (to be analyzed), researchers have to design and implement experimental protocols on their driving simulator. Designing experiments is a quite critical component in any research [5]. It requires careful consideration and controlled environment (depending on the research objectives) to achieve the desired goal of the experiment. Implementation of the experimental protocol includes the specification of subject vehicle, autonomous traffic, simulation environment, and the critical events or scenario modeling. Besides traffic and weather specification, scenario modeling is one of the steps while implementing an experimental protocol, which requires extensive programming, so end-users (behavioral researchers) need specific technical and programming skills for which they are not formally trained.

Modeling scenarios on a driving simulator is a complex and difficult task for the behavioral researchers, because of their complex and technical nature. In most of the cases, end-users are not equipped with the skills to manage these tasks. Also, in most of the existing driving simulators, the

 ISBN: 978-1-61208-250-9 

scenario editors do not take into account the skills of the primary users (behavioral researchers) of driving simulator. They have to depend on the technical persons in their respective organization, who model scenarios. This is a time-consuming task for them because of dependency on the other persons. It is a time-consuming task for technical persons as well. In the case of scenario programming on driving simulators, the key challenge is to make events happen at the right time and at the right place in a non-conspicuous way, which is because of two main factors [6]. First, driving behavior is complicated and not well-understood; so it is difficult to create realistic as well as controllable traffic. Second factor is the variability of human driving behavior, as they change their speed, lane position and tactical decisions with the time during the simulation trial, which leads to variance in drivers behavior to be studied. For example, imagine you want to create an accident situation, where a car overtakes the participant car and stops in front of the participants vehicle. To implement this situation using a modeling language, there are many parameters to control and consider including speed and position of the overtaking vehicle, speed of the participant vehicle and the distance to stop from the participant vehicle. Every driving simulator provides a scripting environment, where behavioral researchers can model scenarios, by using triggers and actions or some event-driven mechanism. Besides the main issue of programming the critical situation without having enough skills, the interaction environment provided by these driving simulators does not help the user to account for the skills they lack in order to perform a task using the driving simulators. So, the objective of our work is fill the gap between the user skills and the goals they want to achieve using driving simulators .In this paper we focus on the analysis of the users, their activities and propose a solution to help them to achieve their goals while not being expert in programming.

The remainder of the paper is organized as follows. The next section describes the related work. Next, the third section introduces, our approach along with the theoretical framework Then, the fourth section provides a description of the evaluation of our approach by the end-users, which is followed by the discussion, conclusion and future work.

## II. Related Work

Significant work has been done on End-User development. End-user activities take user-centered approach to develop tools and systems which can vary from, customization or parameterization, where user can set the parameters to extend or change the system functionality or presentation of the data [7] e.g. Programmorphosis [8], to the modification or the development of computing artifact from scratch e.g. Visual programming, Programming by example [9] etc.

In the rest of the section, we discuss briefly about different flavors of interaction environment for scenario modeling systems and the methodology used by the driving simulators for modeling scenarios. The purpose of our discussion is to highlight the scenario modeling approaches being used in different softwares and not to compare these softwares. We also do not discuss the details of these systems or their software components, because this is not the focus of the paper, we discuss briefly about the User Interface of some of these simulators i.e. how to place scenario objects in the scenario, and how to manipulate the scenario events.

SCANeR software [10], developed by OKTAL SA and Renault has a Graphical User Interface (GUI) for implementing the experiment protocol and to model scenarios. The scenario objects (vehicles, traffic signals..) are placed on the map using mouse. The ambient traffic is generated by placing the sources on the map or by placing individual vehicle in the map. To construct critical situations, it uses GUI for defining condition-action pair (If-else statements).

ARCHISIM [11] developed by IFSTTAR has a textual interface to define an experimental protocol and to model scenarios. The scenario objects are specified in the text files by specifying the location on the road. Ambient traffic is created manually using text files. To construct critical situations, it uses a simple text editor like notepad. In the notepad, the user can specify the condition action pair for different events.

STISIM [12] is developed by System technology, Inc. It has a textual interface for scenario modeling process. SDL (Scenario Definition Language) is a scripting language developed to define the scenario event. The scenario objects are placed in the scenario by the route traveled by the driver using SDL statements. As the participant drive in the world, the objects (buildings, traffic signals...), critical events and ambient traffic is defined using SDL.

Besides the above mentioned interfaces to model scenarios, some other simulators use different methods for scenario modeling. Wasshink et al.,[13] has proposed a movie set metaphor to generate scenarios dynamically based on Green Dino Virtual Realities Dutch Driving Simulator. They have proposed the movie set as a driving simulator, where actors (vehicles, pedestrians etc) come at the scene and play certain set of roles, which are assigned to them in the script. They have also emphasized on the problem of users to model scenarios using a scripting language. A tile-based approach is also used to specify scene and scenario elements in the driving simulator. The whole world (Terrain/Map where participant drive the car) is divided into different tiles, which are configured, assembled and then loaded into the driving simulator during the experimental trial. A Tile is a section of the route on which contains the elements like roads, traffic signals, buildings, trees and scenario objects. These tiles are then grouped together and loaded using an interface or by specifying the sequence of the tiles. The scenario events are then created on the map. In some systems, tiles are static and may not be altered or moved during the

 ISBN: 978-1-61208-250-9 

experiment run [14], while there are some systems in which tiles as well as data on the tiles (Scene objects, scenario objects) can be altered dynamically during the experiment run [15]. Sometimes it is required to generate a script dynamically during the simulation. Some driving simulators provide this functionality, and some use other means to fulfill this requirement. All these systems and methods have not been designed and developed by keeping in mind the skills of the primary users (behavioral researchers) of the driving simulators. Also, interaction environment of these systems is not so user-friendly that they could account for user skills. So there is a need for a user-centered interaction environment, which could enable users to program scenarios without having extensive programming skills.

## III. METHOD

In this section, we present the methodology that we have followed to achieve our goals. As we are following the UCD approach. we have involved the users at the very start, and conducted a survey of primary users of driving simulators. After that, we have proposed and designed the solution and took initial feedback from the users.

### A. User Survey

A user survey was conducted to get to know about the users problems and ideas to model scenarios on driving simulators. We interviewed 19 driving simulator users with various backgrounds using different simulators. Most of the users we interviewed were using ARCHISIM and SCANeR softwares. The focus of the interviews was to discuss about their problem, needs and requirement while modeling scenarios on driving simulator. For detailed methodology and results see [16]. In this survey, users have explained their problems and have given ideas about the interface of driving simulator which could fill the gap between their skills and the goal they want to achieve, regardless of any software.

*1) Main problems identified:*

1) Controlling the ambient traffic around the participant during the critical event.
2) Tuning or optimizing the critical events.
3) Finding relevant functions in the scripting language to perform a task.
4) Optimizing and debugging the script.
5) Selection of triggers (time or distance) to model a critical event.

*2) Main ideas proposed:*

1) Drag and drop the critical events/situations using the mouse. These critical events should be editable at low-level using the low-level scripting language of the driving simulator.
2) Interaction of the users with the map. Users want to interact with the map while modeling the scenarios.
3) Preview of the activity they are performing to design a complete experimental protocol.

In our survey, 7 out of 19 users had no programming experience at all, and they had never programmed scenarios by themselves while implementing an experimental protocol. 9 out of 19 users had very little programming knowledge. So they have to take total or partial help from the technical persons while modeling scenarios. The users had an average 2 years experience of working on driving simulators.

### B. Proposed Approach

Every existing scenario modeling system provides a mean to interact with the scenario modeling software, which is textual or GUI. But these interfaces still do not fill the gap of user skills and goals. Based on the discussions with end-users during the survey, we identified the following steps which end-users undertake while designing and implementing an experimental protocol.

1) Terrain selection
2) Configure participant/subject (Initial position, ADAS, speed and other platform dependent parameters).
3) Configure the ambient traffic (Number of vehicles per hour, their configuration i.e. speed, itinerary...).
4) Configure the environment (weather condition, light..).
5) Set the dependent variables to be collected.
6) Construct critical events using scripting.
7) Experiment execution and data collection (variables).

We propose a new User Interface (UI) and experiment development approach based on the problems raised and the ideas proposed by the users during the survey and the aforementioned steps. Traditionally, in order to implement an experimental protocol, a user (technical person or researcher) uses the same interface for implementing an experiment protocol, regardless of level of their technical and programming skills.

In the proposed design we split the scenario modeling activity into 3 sub-interfaces depending on the roles they have to perform while modeling scenarios and the set of skills they have. The 3 roles ”Researcher” as ”R1” (Low or no programming skills), ”Technical Person” as ”R2” (Good programming/technical skills) and Operator as ”R3” (No technical skills required) correspond to the Experiment Builder, Template Builder and Experiment Interface, respectively.

We explain this new approach and interface with the help of an example. Our example scenario contains two events.

- **Accident Event:** A vehicle overtakes the participant vehicle by increasing its speed and changes its lane to the lane of the participant vehicle and then brakes.
- **Pedestrian cross event:** A pedestrian walks and then crosses the road as the participant vehicle approaches.

*1) Template Builder:* This sub-interface will be used by technical persons performing role R1 having good programming and technical skills. R1 will design the GUI-based templates of the scenario events. The template builder

 ISBN: 978-1-61208-250-9 

will let R1 use existing functions offered by the scenario-modeling environments of the driving simulator to model scenario events. In our example, at the back-end, for the template ”Accident”, the ”Template Builder” will let R1 program a vehicle around the participant vehicle to accelerate, change position, and brake at some distance from the subject vehicle. For the event ”Pedestrian crossing”, the ”Template Builder” will let R1 program a pedestrian to walk and cross the road as the participant vehicle approaches the intersection. At the front end of the template, there would be different text fields to specify the parameters for the events ”Accident” and ”Pedestrian crossing”. These parameters will be filled by the Researcher in the ”Experiment Builder” sub-interface, if he or she wants to modify the default values. The templates developed by R2 will be stored in a template library, so that researchers could access the templates in the ’Experiment Builder’ interface as described in Figure 1.

*2) Experiment Builder:* This sub-interface will be used by researchers/trainers performing role R2 and possibly have low or no programming skills. R2 will define the whole experiment using a user-friendly and intuitive GUI which includes specifying experiment condition, environment, ambient traffic, and data to be collected. To specify the critical events or situations to be studied, R2 will access the template library developed by technical persons in the ’Template Builder’ and will place them in the scenario editor using drag and drop and proceeded by a user-defined trigger. If needed, R2 will fill the parameters of the template.
In our example, end-user will specify the position and actors involved in the templates ”Accident” and ”Pedestrian crossing” besides the template parameters, if needed. There will always be default appropriate values for all the parameters of the template.

*3) Experiment Interface:* The experiment interface will be used by user (researcher or the person who will execute the experiment, depending on the organization culture) performing role R3. Using this sub-interface, R3 will load and execute the scenarios in the driving simulator, developed with the ”Experiment Builder”. R3 can change the parameters of the scenario or template (if needed), during the experiment trial and finally will collect the data to be studied. We will be focusing more on sub-interface ”Experiment builder”, because researchers are our focused users, who do not usually have programming skills. So in the next section we will be talk about our prototype building experience of ”Experiment Builder”.

### C. Prototype development

A prototype based on this concept is developed. In this paper, we are focusing on the Experiment builder sub-interface, which will be used by researchers. We do not discuss the solution in detail because of the space constraints, but we try to present the transformation of user requirements into feature to fill the skills gap of the researchers, which is

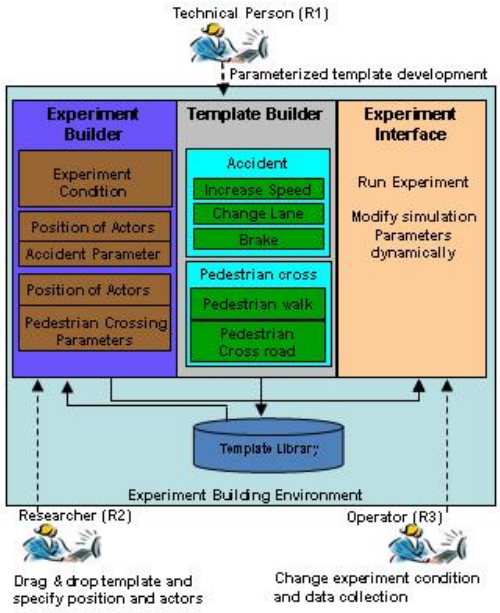


Figure 1. Scenario Modeling Process

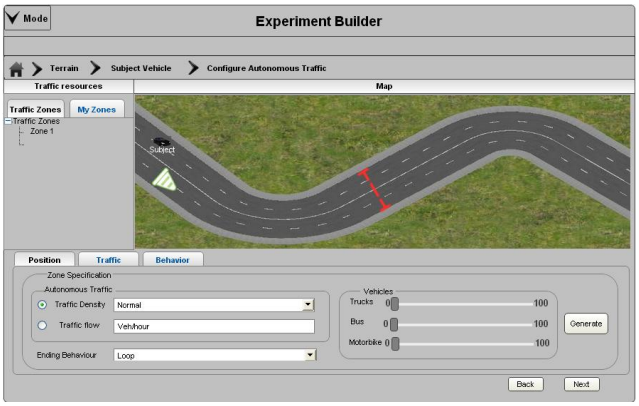


Figure 2. R2 specifying Autonomous traffic

the focus of this article. The prototype is based on the problems identified, user suggestions and the steps that are identified from the user survey. The prototype is built using ”Justinmind Protyper” [17], a tool to develop rich interactive wire frames. It is difficult to explain all the steps because of space constraints, but we explain the main steps proposing solution to user problems. The user is guided using the Breadcrumb navigation [18] during the experiment building process, as you can see in Figures 2, 3 and 4. Targeting problem 1 in User survey section, the end-user can specify the autonomous traffic at higher level by specifying and configuring the traffic zone, rather than scripting individual vehicle as specified in the Figure 2. Targeting problems 2, 3, 4 and 5, a user can create critical events as specified in the Figure 3.The user will just drag and drop the template in the scripting area and customize the template. By customization,

 ISBN: 978-1-61208-250-9 

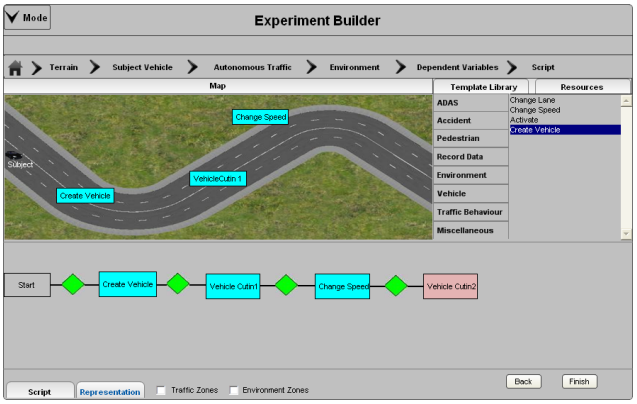


Figure 3. Top: 2D representation of map and resources. Bottom: R2 Speciying critical events using drag and drop

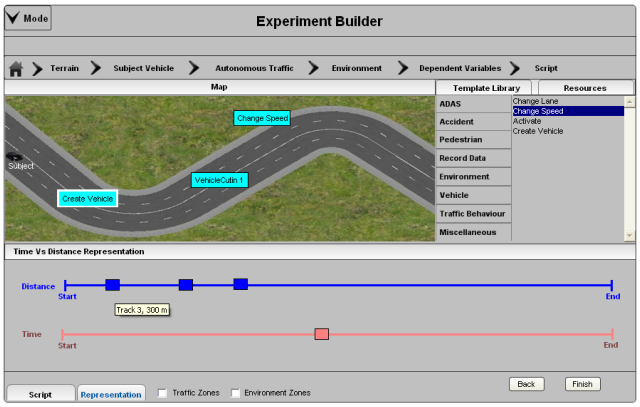


Figure 4. Top: 2D representation of map and resources. Bottom: Temporal and Spatial representation of the experiment

users can easily optimize and tune the critical events. We have also tried to incorporate the user ideas of interacting with the map, i.e. during the creation of environment zone in the environment step, during the creation of traffic zones in the autonomous traffic step and at the last step, where the user creates the critical events. So users can drag the templates from the template library and drop them on the map or in the scripting area. The light blue templates on the map indicate that they are configured based on position of the participant vehicle, and the pink one's indicate that they are configured based on simulation time (See Figure 3).

The division of experimental protocol in different steps enables user to create the experiment in the way they imagine, because traditionally, they have to perform all the steps of experimental protocol at one place, which is fuzzy to tackle, if the scenario is long and complex. In order to keep the scripting process transparent for the users, the whole experiment can be viewed as temporal/spatial representation as specified in the Figure 4.

### D. Prototype Evaluation and Results

Feedback was collected from users about our prototype. It was qualitative evaluation, as it was the initial phase of the design, and we were interested to discuss the issues in detail for a subjective evaluation. Nine users were involved in this evaluation. These users were the primary users of the driving simulators who design and implement the experimental protocols on driving simulators and have low or no programming skills. The approach was explained to the user and then they performed a small exercise on the prototype. Users were observed during the exercise and interviewed later on. The interviews were not recorded and average time for a user was 30-40 minutes with 5-7 minutes for explanation, 15-20 minutes for the exercise and 10-20 minutes for the interviews. In the interviews, users were asked about their feedback with respect to the new approach, division of experiment protocol steps, the difficulties during the exercise i.e. which step was difficult to comprehend, and they were also asked that, is there anything that needs to be improved. The questions were not limited to what we have specified or prepared. We discussed in details, if users had raised any issue. In the exercise, the users created a small scenario, in which first, they had to create a lead vehicle to follow, which was followed by a critical situation in which they had to enable a vehicle to cut-in the participant vehicle. After this event, there was another event, in which the vehicle being followed brakes. And in the last there was another cut-in situation as shown in the Figure 3.

The structure of UI and the steps to create the experimental protocol were quite clear to all the users. They all gave a quite positive feedback about the approach for creating scenarios/events by using templates. There were some minor problems, for example, 4 users told that creating a zone for autonomous traffic was not very intuitive and a bit uncomfortable, and that was observed for 2 more users. Three users also suggested that, in order to create events based on position, they should be able to drop the template on the map. Before they had to drop the template in the scripting area and later they were able to change the position of the template by dragging it on the map. Five users attempted to do that as well. During the interviews, they felt easy and intuitive to configure the templates, rather than doing the low-level coding, and they were also interacting with the map, so they were aware of the simulation activity that where and what is going to happen.

## IV. Discussion and Conclusion

We have discussed in detail about the user-centered design to develop experimental protocols on driving simulators. In this paper, we have focused on one class of end-users (behavioral researchers), who are the primary users of the driving simulators. The objective is to fill the gap between user skills and the goals they want to achieve in an effective way using driving simulators. We have separated the role

 ISBN: 978-1-61208-250-9 

of technical person, who will implement the core logic of the critical events in an abstract way in the form of event templates using low-level language provided by the driving simulator. In this way, end-users will not have to go through a typical programming process. All they have to do is just to drag and drop the templates from the template library and configure them. It will reduce their time in programming and also the dependency on the technical persons. If the researchers have enough skills and want to develop complex situations by themselves, they can change their role into technical persons and use the Template Builder sub-interface to edit a template or create a new template by themselves. During the user survey, users specified that, making some changes in an existing scenario would be easier for them than writing the complete scenario from scratch.

Another aspect of our solution is the division of the implementation task of experimental protocol procedure into meaningful subtasks. In existing systems, users have to use low-level languages to configure each step in the scripting environment. So it is difficult to tackle complex and long scenarios, even if some users have good programming skills. We have split the whole experimental protocol development procedure into different steps, and have provided support to the users during each step by keeping in mind the skills of the users, which make our approach different from the existing approaches.

Now, we have evaluated the concept and design at its initial phase, We also plan to conduct a controlled experiment to evaluate the solution in detail. After the user evaluation, we have improved the user-interface, and now we are implementing the complete system. As driving simulators vary based on their execution platform, so we need to generalize this solution for different platforms, and we are also working on that at the moment.

## ACKNOWLEDGMENT

This work has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement no. 238833/ ADAPTATION project. www.adaptation-itn.eu

## REFERENCES

[1] A. J. Ko, R. Abraham, L. Beckwith, A. Blackwell, M. Burnett, M. Erwig, C. Scaffidi, J. Lawrance, H. Lieberman, B. Myers, M. B. Rosson, G. Rothermel, M. Shaw, and S. Wiedenbeck, "The state of the art in end-user software engineering," *ACM Comput. Surv.*, vol. 43, no. 3, pp. 21:1–21:44, April 2011.

[2] A. Cypher and D. C. Smith, "Kidsim: end user programming of simulations," in *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 27–34.

[3] A. Newell and S. K. Card, "The prospects for psychological science in human-computer interaction," *Hum.-Comput. Interact.*, vol. 1, no. 3, pp. 209–242, Sep. 1985.

[4] J. F. Pane, B. A. Myers, and L. B. Miller, "Using hci techniques to design a more usable programming system," in *Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02)*, Washington, DC, USA: IEEE Computer Society, 2002, pp. 198–.

[5] R. A. Romano and H. A. Stoner, "A quick and effective prototype experimental design and analysis tool," in *Driving Simulation Conference North America*, Iowa, IA, 2007.

[6] K. Joseph and G. Timofey, "Scenario authoring," in *Handbook of Driving Simulation for Engineering, Medicine, and Psychology*. CRC Press, Apr 2011.

[7] H. Lieberman, F. Patern, M. Klann, and V. Wulf, "End-user development: An emerging paradigm end user development," in *End-User Development*, ser. Human Computer Interaction Series. Springer Netherlands, 2006, vol. 9, pp. 1–8.

[8] A. Ioannidou, "Programmorphosis: a knowledge-based approach to end-user programming," in *INTERACT*, 2003.

[9] A. Cypher, "Eager: programming repetitive tasks by example," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '91. New York, NY, USA: ACM, 1991, pp. 33–39.

[10] G. Reymond, A. Heidet, M. Canry, and A. Kemeny, "Validation of Renault's dynamic simulatorfor Adaptive Cruise Control experiments," in *Driving Simulation Conference DSC*, Sep. 2000.

[11] S. Espie, F. Saad, B. Schnetzler, and F. Burlier, "Microscopic traffic simulation and driver behavior modeling: the archisim project." in *Strategic Highway Research Program and Traffic Safety on Two continents*, Lille, France, 1994.

[12] G. D. Park, R. W. Allen, and J. T., Rosenthal, "Flexible and real-time scenario building for experimental driving simulation studies," in *CHI 2011*, Palo Alto, CA, 2011.

[13] I. Wassink, B. van Dijk, J. Zwiers, A. Nijholt, J. Kuipers, and A. Brugman, "In the truman show: Generating dynamic scenarios in a driving simulator," *IEEE Intelligent Systems*, vol. 21, no. 5, pp. 28–32, Sep. 2006.

[14] Y. Papelis, O. Ahmad, and G. Watson, "Developing scenarios to determine effects of driver performance: Techniques for authoring and lessons learned," in *Driving Simulation Conference North America*, Dearborn, Michigan, 2003.

[15] P. Suresh and R. R. Mourant, "A tile manager for deploying scenarios in virtual driving environments," in *Driving Simulation Conference North America*, Orlando, Florida, 2005.

[16] G. Bhatti, R. Bremond, J.-P. Jessel, G. Millet, and F. Vienne, "User requirements to model scenarios on driving simulators," in *5th International conference on Driver behaviour and Training (ICDBT)*, Paris, France, 2011.

[17] "Justinmind prototyper." [Online]. Available: http://www.justinmind.com/ (Accessed: 29/01/2013 )

[18] J. Nielsen, "Breadcrumb navigation increasingly useful," in *Jakob Nielsen's Alterbox*, April 10, 2007. [Online]. Available: http://www.nngroup.com/articles/breadcrumb-navigation-useful/ (Accessed: 29/01/2013 )

 ISBN: 978-1-61208-250-9 