HAL
archives-ouvertes.fr

A detailed description of a user-centered interface to model scenarios on driving simulator

Ghasan Bhatti, Guillaume Millet, Jean-Pierre Jessel, Nguyen-Thong Dang, Fabrice Vienne, Roland Brémond

▶ To cite this version:

HAL Id: hal-01144895

https://hal.archives-ouvertes.fr/hal-01144895

Submitted on 23 Apr 2015

A Detailed Description of User-Centered User-Interface to Model Scenarios on Driving Simulators.

Ghasan Bhatti[1], Guillaume Millet[2]

Research Engineer/Ph.D Student[1], Head of Software Development – Automotive[2]

OKTAL SA

19 Boulevard des Nation Unies, 92190, Meudon, FRANCE

{ghasan.bhatti, guillaume.millet}@oktal.fr

+33 (0)1 81 66 65 56[1], +33 (0)1 46 94 97 84[2]

Jean-Pierre Jessel

Professor

IRIT, Paul Sabatier University, Toulouse

118 Route de Narbonne, F-31062, Cedex 9, Toulouse, FRANCE

jean-pierre.jessel@irit.fr, +33 (0)5 61 55 63 11

Nguyen-Thong Dang[3], Fabrice Vienne[4], Roland Brémond[5]

Research Engineer[3], Head of Virtual reality team[4],Senior Researcher[5]

LEPSiS/IFSTTAR, Paris-Est University

14-20 Boulevard Newton Cité Descartes, Champs sur Marne F-77447

Marne la Vallée Cedex 2, FRANCE

{nguyen-thong.dang , fabrice.vienne, roland.bremond }@ifsttar.fr

+33 (0)1 30 84 39 40[3], +33 (0)1 81 66 65 86[4], +33 (0)1 81 66 65 34[5]

ABSTRACT

Modeling scenarios on driving simulators is a complex and difficult task for end-users because most of them usually don't have the skills required to program the scenarios. In this paper, we present a User-centered solution in which we split the scenario modeling interface into 3 sub-interfaces (Template Builder, Experiment Builder, Experiment Interface) based on user skills. We have developed a prototype of the interface, which is explained in detail.

Keywords: Driving Simulators, Scenario modelling, User-Centered Design, Experimental Protocol, User Interface

INTRODUCTION

Driving simulators are increasingly used by end-users (e.g. psychologists, human factor experts etc) to study driving behaviors, road safety features and to design and evaluate Advance Driving Assistance Systems (ASAS). In order to carry out an experiment, they have to prepare an experimental protocol, which involves scenario authoring (for example, modeling traffic situations and vehicle maneuvers, etc.). Such a preparation requires technical and programming skills for which behavior researchers don't have any formal training in most of the cases. As a result, they have to depend on technical persons or scenario developers in their respective organizations, which is a time-consuming process because of the dependency of on other persons.

Scenario means a sequence of actions or events in a general term. In the context of driving simulators, scenario can be regarded as "Everything that happens in the driving simulator", which includes specifying and controlling ambient traffic and its attributes, ambient environment and simulation conditions, route of the participants and their position, and the traffic situation and vehicle maneuvers Papelis (2003). It is sometimes used to specify both the layout and the activities during the experimental trial. Some authors use the term "Scene" to specify the layout (road network, terrain, driving environment e.g. city, motorway etc) and the term "scenario" to specify the activities (critical situations, vehicle maneuvers etc) during the experimental trial Kearney and Timofey (2011). In our context, scenario means to specify the activities including critical events, vehicle maneuvers, environment change etc.

In order to develop scenarios, end-users take technical and programming help from the technical persons and scenario developers. There is also a communication gap between the end-users and technical persons, as there is no standardized language or method for communication between researchers and technical persons.

One of the reasons why programming is a difficult task for end-user is that programs are abstract. Cypher and Smith (1995). It is difficult for end-users to think in an abstract manner to implement the specific task rather than thinking about the situation in the real-world.

Every driving simulator provides a scripting language and the scenario authoring tool to develop scenarios. These scripting languages are powerful enough to develop any kind of driving scenario but they do not take in to account the skills of the end-users and are at quite low-level that they could account for these skills. Also the programming languages have not been designed by addressing the issues of Human-Computer Interaction (HCI) Newell and Card (1985). According to Pane et al. (2002), user interface is one of the factors which can make

programming a difficult activity for the end-users. The End-User Development (EUD) systems should fulfill the goals of end-users by taking their skills and objectives into account.

The interaction environment of driving simulators is also not free from the issues described above. Scenario modeling activity is a challenging task for end-users because of two main factors. First, driving behaviour is complicated and not well-understood, so it is challenging to create a reproducible and realistic traffic. The second factor is the variability of the driving behavior.

So, there is a need to develop a standardized and user-centered scenario authoring environment that enable and empower the end-user to develop the scenarios using the skills they have. In this paper, we present a detailed user-interface (UI), that we have developed by adopting the User-Centered Design (UCD) approach. The remaining sections of the paper are organized as follows. In the next section, we present the related work, which is followed by the details of our method. Then we propose the new interface and its evaluation, which is followed by a conclusion.

RELATED WORK

Significant work has been done to develop usable programming and authoring system for the end-users in the past, for example Agentsheet Ioannidou (2003), Alice Conway (2000), Programming by Example Cypher (1991) etc. Existing driving simulation platform differ in the way critical events and the traffic is specified. They provide different user interfaces for scenario authoring which range from textual to graphical format.

SCANeR Software Reymond (2000) developed by OKTAL SA, uses a graphic way to model scenarios. Scenario objects (Vehicles, Traffic signal etc) are placed using mouse and scenarios are modeled using Condition-action pair (If-else statements) in the authoring environment.

ARCHISIM Espie et al. (1994), developed by IFSTTAR, uses textual statements to specify scenario objects and critical events. described in a text editor like 'notepad' to specify scenario objects and critical events. Users can use a text editor like 'notepad' to prepare their scenario where Condition-Action pair If-else statements in textual format. The traffic and the critical events are specified in different text files.

STISIM Park et al. (2011) developed by System International, uses textual statements in their Scenario definition language (SDL). Scenario objects and critical events are described using SDL by the route travelled by the participant.

Wassink et al. (2005) 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 world (scene) is divided in tiles, which are configured, assembled and then loaded into the driving simulator during the experimental trial. A 'Tile' is a section of the route which contains elements like roads, traffic signals, buildings, trees and scenario objects. These tiles are then grouped together and loaded using an interface or specifying the tile sequence. In some systems, tiles are static and may not be altered or moved during the experiment run Papelis (2003), 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 Suresh et al. (2005).

METHOD

We have used the user-centered design (UCD) approach to build an interface for scenario moeling. UCD methods claim to provide an end-product, which satisfies the users and enable them to achieve their goal while taking their profile into account. For this, we conducted a user-survey, and proposed a new approach which is evaluated.

User Survey

We interviewed 19 driving simulator users with different profile and experience on driving simulators. For detail about the survey see Bhatti et al. (2011). During the user-survey, users have explained their problems and suggestion about the use of driving simulators in detail. A brief detail about the survey with common problems and suggestions is written as follows.

User Problems
1. Control ambient traffic around drivers during the critical situations.
2. Tuning or optimizing the critical events.
3. Finding relevant functions to construct an event.
4. Optimizing and debugging the situations.
5. Selection of triggers to model a critical event.

User Ideas
1. Drag and Drop the critical situations at high-level at the specific place on the scene.
2. Interaction of users with the scene.
3. Preview of the activities to design an experimental protocol.

From the User-Survey, we have also identified the steps, that end-user follows in order to develop an experimental protocol. The steps are:
1. Terrain/Scene selection.
2. Configure participant vehicle (position, ADAS…)
3. Configure the autonomous traffic.
4. Configure Environment (i.e. daylight, weather..)
5. Select independent variables of experiment.
6. Construct critical events.
7. Experiment execution and data collection.

The users had an average of 2 years experience of working on driving simulator. Seven out of 9 users had no programming experience at all. And 9 users had a very little programming knowledge. So these users had always take help (total or partial) from the scenario developer to model scenarios.

Proposed Approach

Based on the user survey, we have taken into account the skills of the users and proposed a new User-Centered multilayer programming solution. We have identified three kind of users who

work with the driving simulators at different levels:
1. Researchers/Trainers
2. Technical persons/Scenario Developers
3. Experiment Operators.

Multi-layer programming

In the multi-layer programming solution, end-users (researchers/trainers) interact with the higher-layer, which is close to the real world, while technical persons interact with the middle layer and the Driving simulator manufacturers interact with the lower layer as shown in the Figure 1.

Researchers

Higher-Layer

Middle-Layer

Technical Persons

Lower-Layer
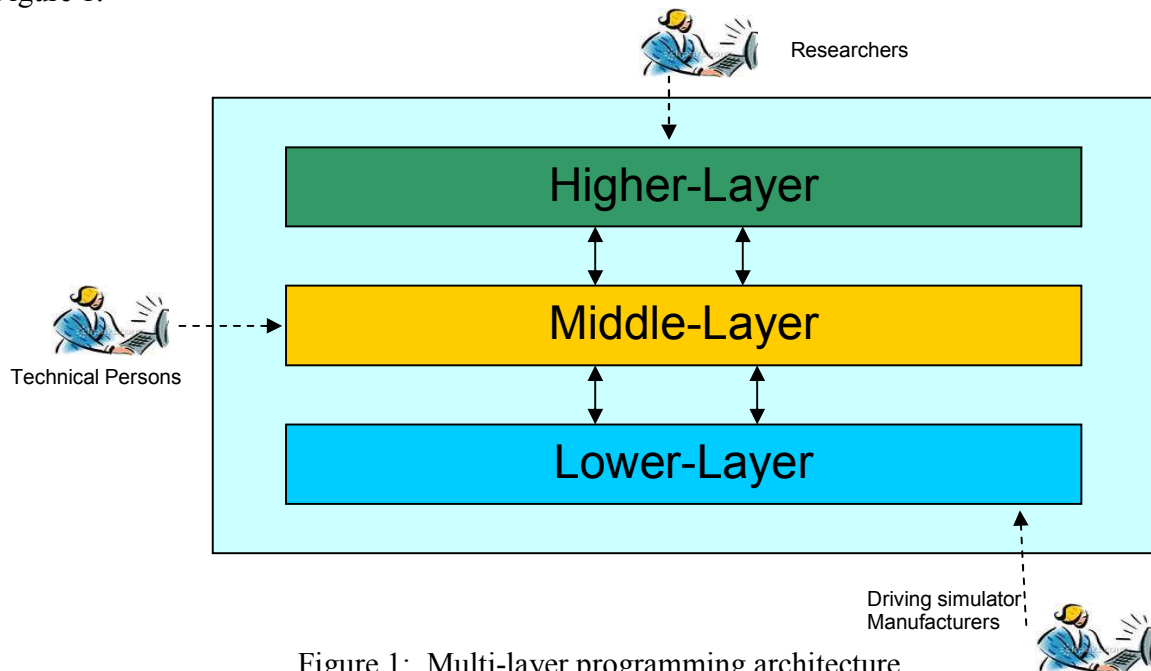
Driving simulator
Manufacturers

Figure 1: Multi-layer programming architecture

Traditionally Researchers and technical persons both interact with the middle layer, and develop the scenarios using the API (Application programming Interface) or the high-level functions (for example to change the environment condition, to change the speed of the vehicle etc) developed by the software manufactures. API is the set of programming instructions to access software applications. These APIs can be developed using low-level programming languages (for example C++ or PHP) by driving simulator manufacturers (for example SCANeR), or they can be developed by technical persons (For example ARCHISIM), if the simulator is built within a research institution. While using APIs for the scenario development, end-users still have to follow the syntax and the low-level programming primitives which is difficult for them to follow in most of the cases.
The APIs are still at lower level and not close to the real world situations for the researcher, so they still have the difficulty in developing scenarios using the APIs as middle layer. We add an extra layer (higher layer) with which the user can develop the scenarios by exploiting the low-level programming primitives and the syntax of the APIs.

Development of the User Interface

As mentioned above, we have identified three types of users who interact with the driving simulators. We split scenario authoring interface into three sub-interfaces corresponding to the three groups of users identified during the survey: Experiment Builder, Template Builder and Experiment Interface as shown in Figure 2. For details see Bhatti et al. (2013).
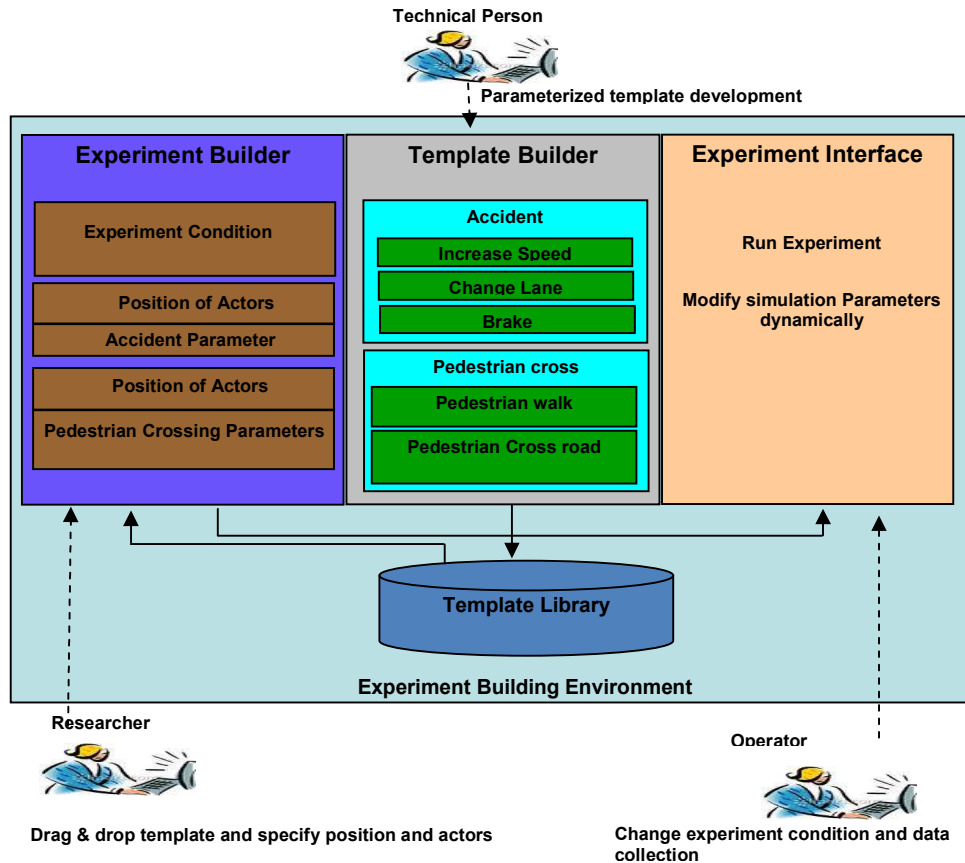

Figure 2: Experiment Building Procedure

The brief description of 3 sub-interfaces is written as follows.

Template Builder

This sub-interface will be used by technical persons who will design the Graphical Interface (GUI)-based parameterized templates of the scenario events. They use the existing APIs of the driving simulators to construct the high-level events, e.g. Cut-in Vehicle, Traffic Jam.

Experiment Builder

This sub-interface will be used by researchers who have no or very low programming skills. They will customize the parameters of the scenario events (at High Level) rather than using traditional scenario authoring interface.

Experiment Interface

This sub-interface will be used by researcher or any other person who operates the driving simulators. They run the experiment and can change the parameters of the templates and collect data.

As mentioned above, we propose a multi-layer programming solution. The 'Experiment Builder' corresponds to the Upper layer, while 'Template Builder' corresponds to the middle layer. It is a multi-layer programming solution. Upper-layer is for researchers, where they can customize the parameters of the scenario event templates at high-level with minimum set of skills, and lower layer is for the technical persons who develop the templates at low-level.

During the development of experimental protocol using 'Experiment Builder', users will follow the 7 steps that we have identified during the user survey. Traditionally, users use low-level scripting languages to configure each step in one scripting environment. So it becomes difficult, when the scenario is complex and long. In our solution, we have divided the experimental protocol procedure in to meaningful sub-tasks. These steps are explained in detail in the next section.

Prototype Development

As the focus of our work is on end-users, so we present the prototype for the Experiment Builder. The prototype is built using "Justinmind prototyper", a tool to develop interactive prototypes JustinMind (2013). The user was guided using breadcrumb navigation Nielsen (2007). In the rest of the sub-section, we explain all the steps which will be followed by the researchers.

Step1: The end-user provides the name and description of the experiment.
Step 2: The end-user select the terrain (already developed) from the terrain database as shown in the Figure 3.
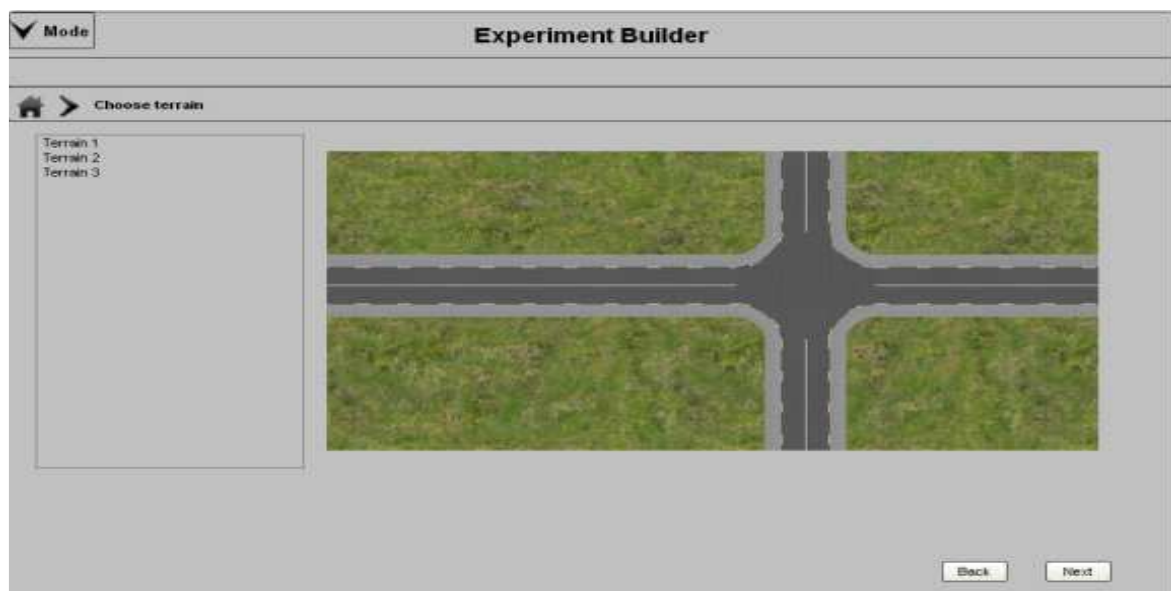

Figure 3: Step 2-Specify Terrain

Step 3: The end-user configures the subject vehicle. He/she provides the vehicle model, color, position, and the maximum speed. The user can also specify the Itinerary of the vehicle, the ADAS and its parameters as shown in the Figure 4.

Figure 4: Step 3-Configure Subject vehicle

Figure 5: Step 4- Configure Autonomous traffic

Step 4: The user specifies the traffic by creating the traffic zones. He will create the zones by specifying the position of each zone on the map. Then he can specify the density of the traffic flow in each zone. He can specify the distribution and the behavior of the traffic in the zone as shown in Figure 5. Traditionally, users specify the traffic by specifying individual vehicles, which is time-consuming in the case of dense traffic, as they have to configure each vehicle and its parameters.

Step 5: The user specifies the environment (Light, Fog, rain, snow). He/she can also specify the environment by creating a zone, if he/she wants to test some condition in rain or in fog. Then he/she can specify the zone position and the environment parameters as shown in the Figure 6.
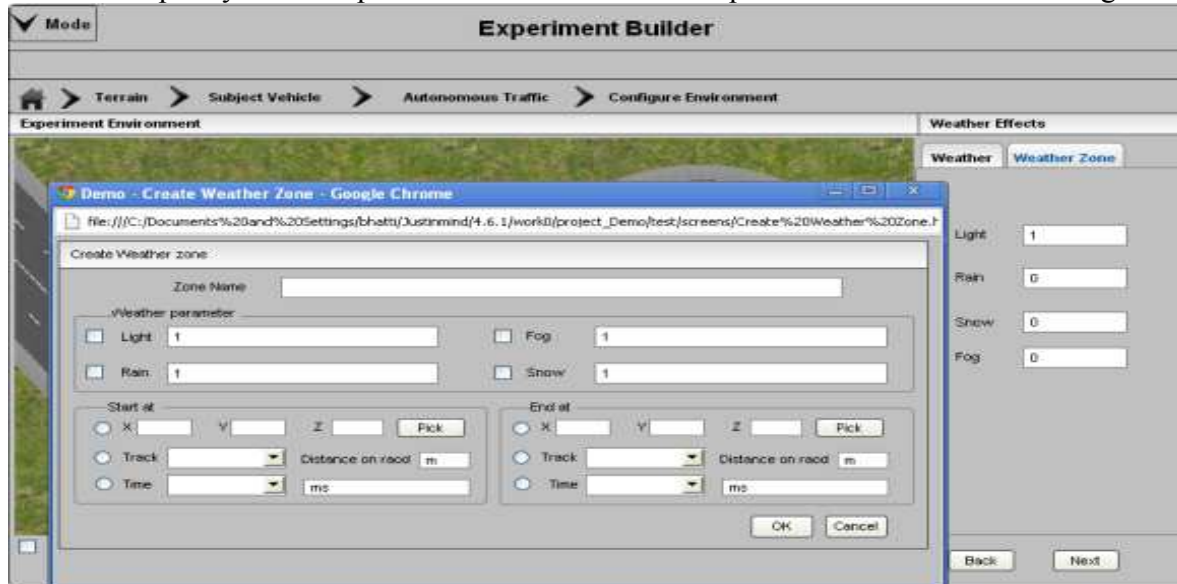

Figure 6: Step5-Configure environment zones.

Step 6: Users specify the dependent variables they are going to measure during the experimental trial as shown in the Figure 7.
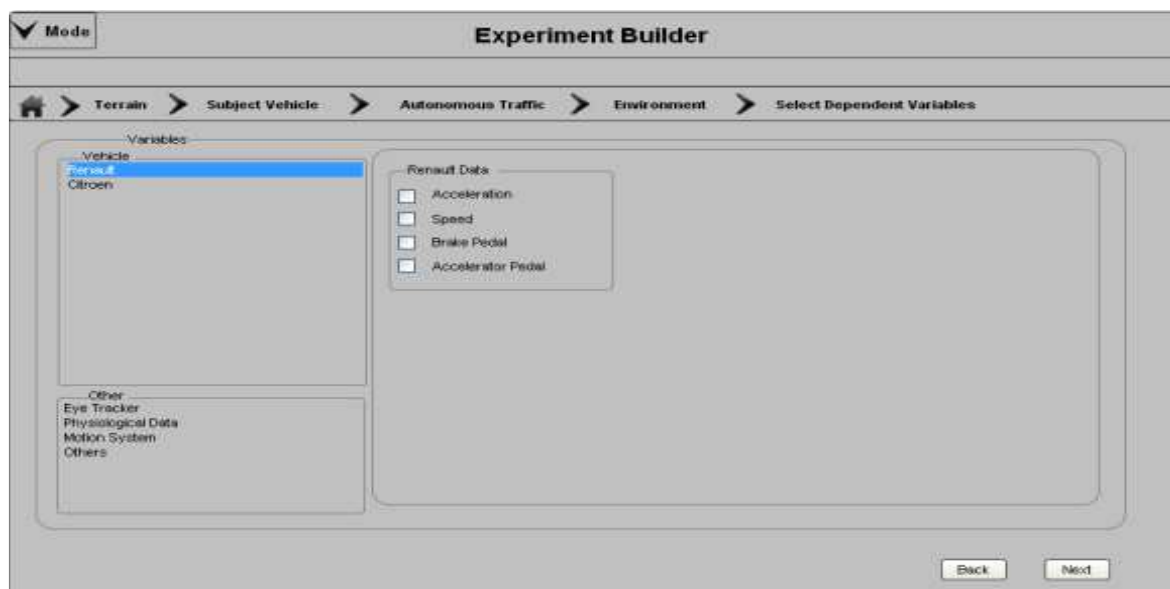

Figure 7: Step6-Select dependent variables.

Step 7: Users have to create the traffic situations and vehicle manoeuvres, which they want to study during the experimental trials. Users drag the template of the traffic situation or the vehicle manoeuvres from the template library to the scripting area and configure the parameters of the template.
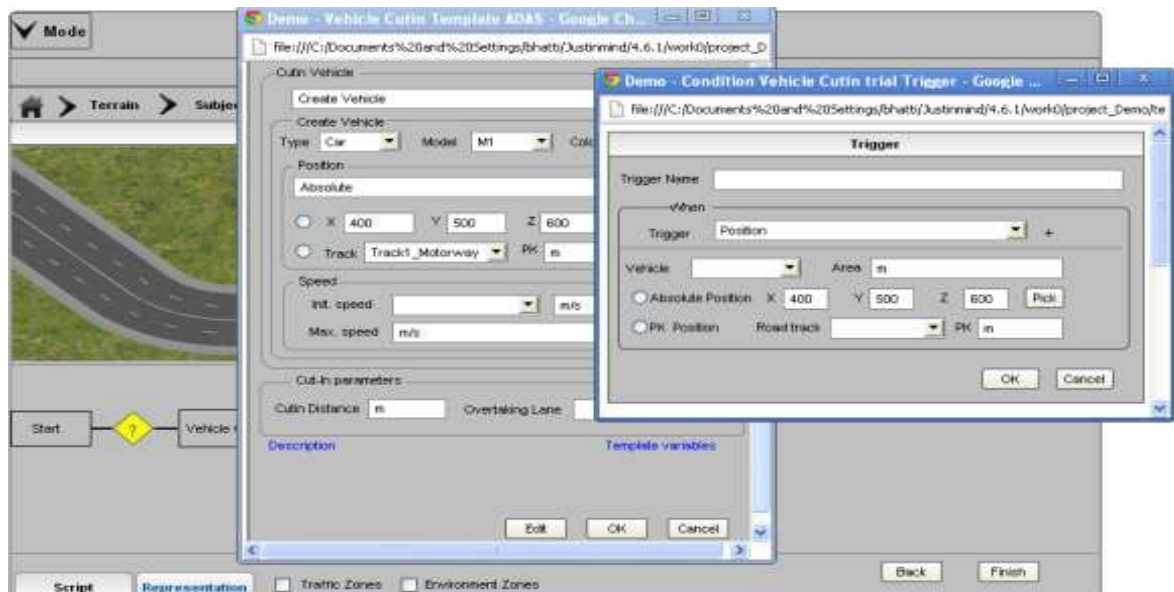
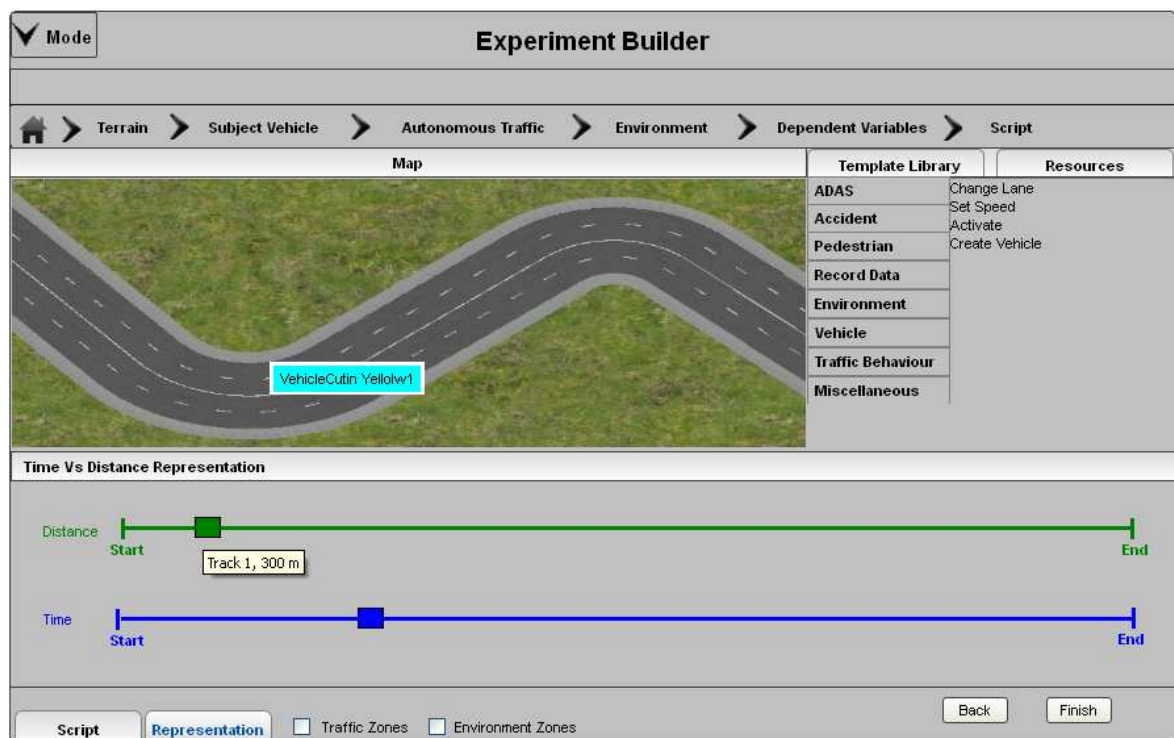Figure 8: Step7-Construct critical events and situations.

Figure 9: Visualize experiment on time-line and distance-line

These templates will be developed by technical persons using the 'Template Builder'. After dragging the template in the scripting area, the user defines the condition by specifying the trigger indicating when this template will be executed (e.g. at a specific position of the subject

vehicle, or at the specific time etc). Then the user can specify the parameters of the template as shown in Figure 8.

Step 8: Users can visualize the whole experiment on a temporal-spatial representation. The templates which will be triggered based on time can be viewed on a time-line, while the position-based templates can be viewed on a distance-line as shown in the Figure 9.

Discussion and User Feedback

We present the UCD solution to model scenarios on the driving simulators, in which we have separated the roles of the user interacting with driving simulators. One of the difficulties in the programming is the division of a programming task into meaningful sub-tasks. We have divided the experimental protocol procedure into different sub-tasks corresponding to the user way of developing the experiment. We have also provided the support at each step using different interaction techniques. The customization of the template will enable the end-user to develop the experiments without following typical programming process.

As we are following the UCD approach, so we have conducted a user evaluation of the prototype. Nine users were involved in the evaluation process. During the evaluation, users performed a small exercise using the prototype. Users were observed during the exercise and interviewed later on. Users have given a positive feedback. The experiment development steps were clear to the end users. They felt easy and intuitive to configure the templates instead of low-level coding. Users have also pointed out some minor problems e.g. for some users, configuring the traffic zones was not very intuitive, and that was observed for some other users as well. For detailed evaluation see Bhatti et al. (2013).

CONCLUSIONS

In this paper we have presented a detailed User-Centered Interface to develop an experimental protocol. As end-users are not equipped with programming skills, so first we have tried to analyze the user problems, and proposed a solution. In this solution, we intent to empower the end-user, so that they could develop the experimental protocol with little or no help from the technical persons. We have also presented the steps in the manner followed by the users to develop an experimental protocol.

After the user-evaluation, the interface is improved and implemented now, and in future we intent to conduct a controlled experiment to analyze our interface in detail.

ACKNOWLEDGEMENTS

REFERENCES

Bhatti, G., Brémond, R., Jessel, J.-P., Millet, G. & Vienne, F. (2011) User requirements to model scenarios on Driving Simulators. In: *5th International conference on Driver behaviour and Training (ICDBT)*, pp. 493-499, Paris, France.

Bhatti, G., Brémond, R., Jessel, J.-P., Dang, N.-T., Millet, G. & Vienne, F. (2013) Filling the user skill gap using HCI techniques to implement experiment protocol on driving simulator. In: 6th International conference on Advances in Computer-Human Interactions (ACHI), pp. 401-406, Nice, France.

Conway, M., Audia, S., Burnette, T., Cosgrove, D., Christiansen, K., Deline, R., Durbin, J., Gossweiler, R., Koga, S., Long, C., Mallory, B., Miale, S., Monkaitis, K., Patten, J., Pierce, J., Shochet, J., Staack, D., Stearns, B., Stoakley, R., Sturgill, C., Viega, J., White, J., Williams, G., and Pausch, R. (2000) Alice: Lessons Learned from Building a 3D System For Novices. *Proceedings of CHI 2000* pp. 486-493).

Cypher, A. (1991) Eager: Programming repetitive tasks by example. *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology* pp. 33-39).

Cypher, A. & Smith, D. C. (1995) KidSim: end user programming of simulations. *Proceedings of the SIGCHI conference on Human factors in computing systems* pp. 27-34). Colorado, USA.

Espie, S., Saad, F., Schnetzler, B., Bourlier, F. & Djemane, N. (1994). Microscopic traffic simulation and driver behaviour modelling: the ARCHISIM project. *Road Safety in Europe and Strategic Highway Research Program*: 22-31.

Ioannidou, A. (2003) Programmorphosis: a Knowledge-Based Approach to End-User Programming. In: Bringing the Bits together, Ninth IFIP TC13 International Conference on Human-Computer Interaction, Zurich, Switzerland.

JustinMind (2013) *Justinmind Prototyper*.

Kearney, J. K. & Timofey F, G. (2011). Scenario Authoring. In: Fisher, D. L., Rizzo, Matthew., Caird, Jeff k., Lee, John D., (Ed) *Handbook of Driving Simulation for Engineering, Medicine, and Psychology*. Boca raton, FL: CRC Press/Taylor & Francis.

Newell, A. & Card, S. K. (1985). The prospects for psychological science in human-computer interaction. *Human-computer interaction* 1: 209-242.

Nielsen, J. (2007). Breadcrumb navigation increasingly useful. *Jakob Nielsen's Alertbox*.

Pane, J. F., Myers, B. A. & Miller, L. B. (2002) Using HCI techniques to design a more usable programming system. *Human Centric Computing Languages and Environments, 2002. Proceedings. IEEE 2002 Symposia on* pp. 198-206).

Papelis, Y., Ahmad, O., Watson, G. (2003) Developing Scenarios to Determine Effects of Driver Performance: Techniques for Authoring and Lessons Learned. *Driving Simulation Conference North America* Dearborn, Michigan.

Park, G. D., Allen, R. W. and T., Rosenthal, J (2011) Flexible and Real-time Scenario Building for Experimental Driving Simulation Studies. *CHI 2011* Palo Alto, CA.

Reymond, G., Heidet, A., Canry, M., and Kemeny, A. (2000) Validation of Renault's dynamic simulator for Adaptive Cruise Control experiments. *Driving Simulation Conference (DSC00)* pp. 181-191). paris, France.

Suresh, P., Mourant, Ronald R. (2005) A Tile Manager for Deploying Scenarios in Virtual Driving Environments. *Driving Simulation Conference North America* Orlando, Florida.

Wassink, I. H. C., Dijk, E. M. A. G. v., Zwiers, J., Nijholt, A., Kuipers, J. & Brugman, A. O. (2005) Bringing hollywood to the driving school: dynamic scenario generation in simulations and games. In: Proceedings of the First international conference on Intelligent Technologies for Interactive Entertainment. Springer-Verlag, Madonna di Campiglio, Italy.