# A Test-Oriented HMI Specification Model for Model-Based Testing of Automotive HMIs

Linshu Duan
AUDI AG
85055 Ingolstadt Germany
linshu.duan@audi.de

**Abstract:** The software functionalities and complexity of today's in-car infotainment HMIs (human machine interface) have grown significantly in recent years. While model-based testing and test automation are widely-used for testing system functions, testing the HMI remains manual tasks, which are very demanding and time consuming. In our research at AUDI AG a model-based testing approach is proposed for testing both the logical behavior and the graphical interface of the automotive infotainment HMIs. As an important part of the testing approach a test-oriented HMI specification model will be designed, which describes the theoretical behavior of the HMI and contains the necessary information for automated HMI testing. HMI tests can be generated from this model. In this paper, we discuss at first the reasons why model-based testing is still not accomplished for HMI testing and introduce a technical solution from a new perspective.

## 1 Introduction

The complexity of HMI software is growing with functionalities and complexity of infotainment systems very strongly [Gmb05]. A user interface giving a faultless experience is one of the most important requirements of today's infotainment systems. However during the development phases, 50 percent of the time and more than 50 percent of the costs are expended for testing [MS04]. While testers of non-HMI applications have been enjoying the convenience of tools to automate their tests, GUI testers still toil in their tedious works. Test designers have to spend a lot of time for defining tests and especially adapting the existing tests for different system variants and software updates. The testers have to execute the tests step by step manually. The reasons why it is so difficult to permute model-base testing are discussed in Section 3. Our technical solution of a new perspective is introduced in Section 3.

## 2 Related Work

Automotive manufacturers and suppliers have busied themselves with the model-based testing of the infotainment system HMI since some years. In [Fle07] a model-based HMI

prototyping tool is suggested from the supplier prospect to manufacturers for a model-based specification. In [SBK05] a similar HMI framework is proposed for model-based HMI specification, development and testing for infotainment HMIs. In [GB09] domain specific languages are introduced to specify the different layers of an infotainment HMI. Model transformation is proposed to be used for integrating these specifications together into one model, which should be able to be used for test generation. But actually, almost all HMI prototyping tools use their own domain specific languages for different layers of an HMI and support visual editors to edit the domain specific languages. A model created by such a tool is already a integrated and complex model of HMI. These existing approaches all focus on HMI frameworks.

## 3 Model-based Testing for Infotainment HMIs

At the moment, the existing approaches in automotive area look for solutions in HMI specification and development frameworks to enforce the model-based testing approach for infotainment HMIs. However, the true reasons are not the only absence of a suitable framework. Firstly political problems exist. Company frontiers and fears of changes in the work process prevents the efforts. Secondly the designer and IT engineer problem, which is fully discussed in [PVH07] are very often the reasons for insufficient quality of the model-based specifications, which is the precondition for model-based testing.
In our approach, we don't face the politic, process or designer and engineer problem directly. In our respects, it is missing a complete model-based testing approach specially identified for infotainment system HMIs, which covers all process parts and is evaluated as practical and usefully. Existing approaches search solutions in the HMI framework. We try to resolve the problem from another perspective. We focus on the content, methods and needed interfaces of the model-based testing approach for infotainment context. Which information we need to automate HMI testing, what kind of tests we need specific for infotainment HMIs, how can tests be generated and executed, which interfaces are need for observation of the SUT behavior are questions we face. In the Section 3.2, a test-oriented HMI specification model is introduced as one of the parts of our solution.

### 3.1 HMI Specifications

HMI specifications are usually created by the manufacturers and describe the requirements in the HMI implementation. Thus, they define the theoretical behavior the implementation has to conform to. Visio and PDF documents are the most widely used format to specified the HMI, although model-based development and specification have been made quantum jump in many other areas. Suppliers performing the HMI implementation promise continuous HMI development process with there HMI prototyping tools [Fle07]. However, the reality is not far from the promise. The working process to create a model-based HMI specification concerns usually more departments with persons coming from different subject areas and requires incessant and interactive corporation between the supplier and


Abbildung 1: Screen and Widgets

the manufacture. Furthermore, the resulted specifications are not satisfying. Specifications and the specification phases have totally different purposes then the implementation. The reuse of elements and efficiency are not focuses of the specification. Instead, simple and understandable descriptions are the main purposes of the specification phases. This different result additional works and insufficient quality of the produced specification models. These models are usually not qualified for the implementation on this base or automated testing. In this paper we don't want to discuss how to resolve the process problem to achieve a qualified model-based specification. We focus on the testing perspective and strive to identify all information needed for automated testing. Certainly, the approach bases on the assumption, that a model-base HMI specification has to be exist, but it tries to base on the minimum requirements in a model-based specification model and does not make any additional demands. The following subsection will introduced the test-oriented HMI specification model.

### 3.2 Test-Oriented Specification Model

A test-oriented specification model is a model, which describes the theoretical behavior of the SUT and contains sufficient information for testing. Depending on the HMI development process, a test-oriented specification model can be created corporately by the HMI designers and test engineers or by the test generation process on the bases of a normal specification and additional information for testing defined by test engineers. In this case the test-oriented HMI specification model does not have to own a visual form as in a specification tool, it could be a composition of information in the memory. We demonstrate at first a normal specification model, and then create a test-oriented specification model directly with a HMI specification tool and show it visually in order to facilitate the introduction.
An HMI specification is usually composed of two parts according to the two layers of HMIs: Statecharts defining the menu behavior (shown in Figure 2) and the graphical elements i.g., screens and widgets defining the HMI looks (shown in Figure 1). Figure 2 shows a strongly simplified cutout from the statecharts of a model-based specification. The statechart cutout describes the simplified menu behavior in navigation feature. Each state in the diagram is a view state, i.g., associated with a defined screen. As soon as the HMI enters a view state the according screen will be displayed. The HMI stands in *Navigation_Main_Screen* when the feature navigation is accessed. As explained, event mapping details of widgets are contained in the graphical layer, but not in the statechart. The first outgoing transition from the left hand side is executed by the event *country*, which is fired

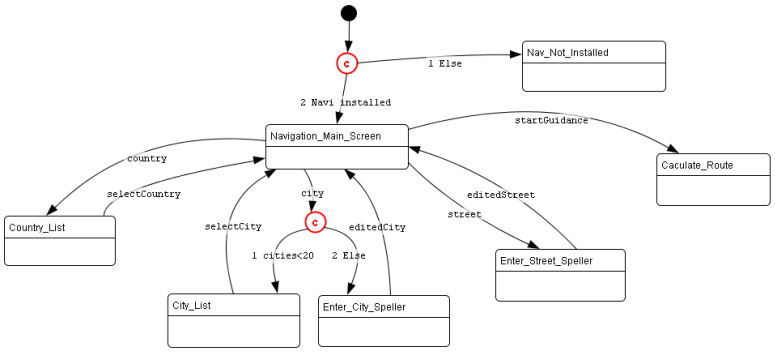


Abbildung 2: Simplified Cutout of an HMI Specification

by the widget country as reaction to the user input through a touch screen or a control panel. In the state *Country_List* the HMI returns to the main screen, when the user has selected a country. From the main screen it is possible to enter a city as destination. If the selected country has less then 20 cities, the screen *City_List* from which the user can directly select the expected city will be entered. Otherwise, the user will be lead to *Enter_City_Speller* which offers a speller for the user to enter the city name. So the next state after the event *city* dependents on the selected country and the database used. By a complete user input the route guidance can now be started. The HMI accesses the screen *Calculate_Route*.

The specification introduced above only defines the theoretical behavior of the HMI. Tests can still not be generated from this specification. In this paper, we want to introduce two mostly important information needed for the testing purpose. Firstly, many important conditions are not accessible for the test generation. The statechart in Figure 2 defines the possibility to reach *City_List* from the main screen. However, the condition to do this is hidden in the widget. The widgets for the specification are usually implemented by the supplier and are available for the specification as libraries. As long as the widget city is active, it fires the event *city* as reaction to the enter event from the user. The widget city is only active when the underlying application, which is not a part of the model-based specification has received a valid selected country and set the widget active. This condition is hidden in the widget library and is not accessible for test generation. Similarly start guidance can only be performed, if the country and a city or a street are given correctly. Without these conditions many invalid tests will be generated. E.g., starting the rout guidance immediately from the main screen without any destination inputs. Secondly contents of user inputs e.g., the city name to be entered do not exist in the specification. Generated tests are not complete without these user input contents.

Figure 3 shows the resulting model after adding the information above into the specification. Actions and conditions are inserted into some of the transitions. E.g., to be able to enter a city the condition *countrySelected==true* must be satisfied. This condition can only be satisfied by the action of the transition from *Navigation_Main_Screen* to *Country_List*. This condition constraints that the country must be selected at first. The conditions will be considered by the test generation introduced in the section 3.3. In this way, it is ensured to

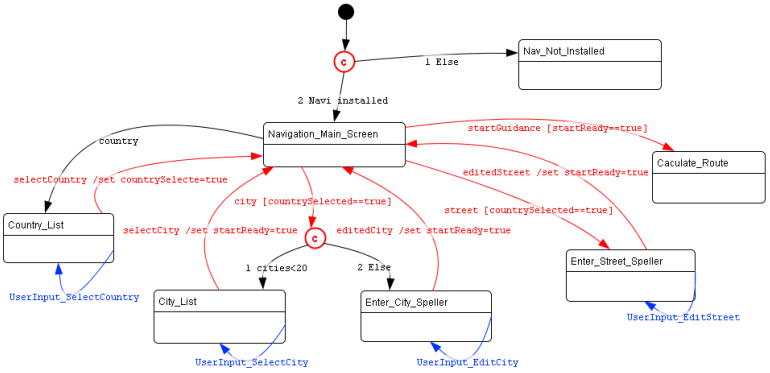


Abbildung 3: Sample of a Test-Oriented HMI Specification Model

generate correct and usable tests and to avoid invalid tests. Some self-transitions indicating the needs for user input contents are inserted into the diagram. Section 3.3 will introduce how user inputs can be integrated into generated tests.

### 3.3 Test Generation and Execution

In order to generate tests form the test-oriented specification, test case specification has to be defined by the testers and a test coverage criterion has to be selected for the forthcoming generation. Test case specification defines the contents and ranges of tests to be generated, i.g., the states and transitions a test should cover. In our work, infotainment system specific coverage criteria are identified. Our previous paper has introduced them in detail.
Before the test generation selects a transition labeled with a condition as the next test step, actions of other transitions have to be determined at first whether make the condition fulfilled. E.g., to select the transition with event *city*, the transition with event *country* has to be selected as previous step. The transitions with *UserInput* events will be selected as steps before all other outgoing transitions from the same state. We explained that the event *country* is the reaction event fired by the widget country. So the test generation selects the action event *enter on widget country* as the test step instead of the reaction event *country*. Generated tests contains both test steps and expected screen or state name.
In the previous paper test tools for infotainment system function tests were introduced. Tests generated from our test-oriented HMI specification model can be executed automatically with the help of such test tools. Test actions performing needed user input contents have to be defined in the test tools before. They will be executed as sub sequence before the rest of the generated tests is executed. Image processing is intensively used to locate the correct widget on the screen and verify whether the HMI stands in the correct menu as expected. Details of test execution, observation and verification will not be explained in this paper to avoid exceeding the paper range.

## 4 Conclusion

The purpose of our research is to find a model-based testing approach for infotainment HMIs. We introduced a test-oriented specification model, which contains both the theoretical behavior of the HMI and sufficient information for testing purposes. Tests can be generated from the test-oriented model and executed with the help of test tools for infotainment system function tests automatically. In future papers, we will introduce details of methods and interfaces executing the generated tests and verifying the SUT behavior. Also solutions for testing the dynamic widget behavior and displaying graphical elements will be introduced. We believe, that we've provided a practical and complete model-based testing approach for infotainment HMIs and other advanced HMIs in this way.

## Literatur


[Fle07] Thomas Fleischmann. Model Based HMI Specification in an Automotive Context. In *Human Interface and the Management of Information. Methods, Techniques and Tools in Information Design*, Jgg. 4557/2007, Seiten 31–39. Springer Berlin / Heidelberg, 2007.

[GB09] Holger Grandy und Sebastian Benz. Specification based testing of automotive human machine interfaces. In *GI Jahrestagung*, Seiten 2720–2727, 2009.

[Gmb05] QNX Software Systems GmbH. Ready for the Future: Software Trends for In-Car Infotainment Systems. 2005.

[MS04] Glenford J. Myers und Corey Sandler. *The Art of Software Testing*. John Wiley & Sons, 2004.

[PVH07] Andreas Pleuss, Arnd Vitzthum und Heinrich Hussmann. Integrating Heterogeneous Tools into Model-Centric Development of Interactive Applications. In *MoDELS*, Seiten 241–255, 2007.

[SBK05] Reinhard Stolle, Thomas Benedek und Christian Knuechel. Model-based Test Automation for Automotive Human Machine Interfaces. Technische Universität Berlin - Forschungsberichte der Fakultät IV, Bericht-Nr. 2005-1, ISSN: 1436-9915, 2005.