Integrated HMI for mobile consumer devices

Reinhard Stolle
BMW Car IT GmbH
80788 München, Germany
`reinhard.stolle@bmw-carit.de`

Daniel Weyl
BMW AG
80788 München, Germany
`daniel.weyl@bmw.de`

INTRODUCTION

Mobile consumer electronics devices, such as mobile phones, smart phones, personal digital assistants (PDAs), music players, are proliferating. The are becoming richer and more powerful in functionality and they are ubiquitously available and usable. A well-designed integration of such devices into the automobile meets the following requirements:

1. a safe and convenient geometric/physical integration of vehicle and mobile devices;

2. functional integration of services that reside on the vehicle's embedded control units (ECUs), on the customer's mobile devices, or on online servers that are available via network portals; exchange of data objects between these services;

3. one consistent, integrated, HMI across all services;

4. HMI behavior that is tailored toward the driver, the current driving situation and the current vehicle state; and

5. integration with intrinsic vehicle functionality such as modern driver assistant systems (DAS) and their HMI.

An integrated solution that enables the integration of mobile CE devices into the vehicle while meeting the above requirements allows the OEM to quickly make new infotainment functionality available in the car without compromising highest quality and safety standards.
In order to make such integrated solutions feasible, several challenges need to be addressed [1]. These include the standardization of interfaces and protocols, extended software architectures with middleware layers that allow for additional abstraction, mechanisms that facilitate the dynamic integration of unknown services, and increased hardware performance requirements. In this talk, we focus on the HMI aspects of these challenges.

HMI INTEGRATION

The minimal extent of device integration concerns physical/geometric aspects: most solutions provide some basic infrastructure, such as a snap-in tray along with some kind of electrical power connection. The next level of integration concerns basic output devices, such as displays and audio speakers. Recently, the trend for those connections has been more and more toward wireless; this is true especially for data transmission (e.g., music streaming), but increasingly also for control functionality (e.g., initiating a phone call). It is not true yet for power connections.

Making such services available via a basic infrastructure is only part of a comprehensive solution. Another, equally important, part concerns the patterns in which the customer interacts with the services, possibly while driving. Therefore, we must pay special attention to providing an HMI that is appropriate for in-vehicle use. Car manufacturers can offer the customer substantial added value by providing a seamless integration of external services and vehicle-based services and offering the resulting functionality to the driver through a comprehensive and comprehensible HMI.

When integrating CE devices into the vehicle, an important question is how the CE functionality can be accessed by the user. In order to leverage the specifically automotive-tailored design of the input elements (buttons, iDrive Controller, speech input) and output elements (displays, iDrive Controller haptics, speech output), the CE functionality must be made accessible through these input/output (I/O) elements. For example, the automotive alphanumeric speller via iDrive is preferable to a small keypad of a mobile phone. However, there is no simple one-to-one mapping between the automotive I/O elements and the I/O elements of CE devices. As a consequence, the mapping of the HMI of a service on a CE

device to the HMI of the automotive environment cannot happen at the level of basic I/O elements. Rather, this mapping must take place at a semantic level of domain-specific functionality and data types.

One of the difficult challenges with the uniform HMI approach is priority management. Since a multitude of services across various devices share a common set of resources (such as displays, audio channels, etc.), we need a framework that allows for the specification of relative priorities of those services.

In addition to the customer requirements discussed above, there are a variety of additional requirements, many of them of legal nature. As described above, the HMI must be appropriate for driving situations and must avoid distracting the driver from the main task of driving. Other legal requirements are concerned with the rights to the content of audio, image and video files that are being displayed on the vehicle's HMI infrastructure.

LIFE CYCLE DECOUPLING

The main technical challenge concerning HMI integration revolves around the following seeming dichotomy: on the one hand, we aim for a life cycle decoupling of CE devices from the vehicle. On the other hand, we aim for a tight integration of the services and of the HMIs in order to leverage the automotive-specific added value. Our goal must be to meet both of these requirements.

Decoupling of life cycles of vehicles and CE devices means that it must be possible to offer the automobile customer an integration of CE devices within a short time after their market introduction. As an immediate consequence of this requirement, the verification phase of integration solutions must be significantly shorter than the verification phase of the vehicle itself. In order to enable this kind of local verification, we need a modular system architecture, which ensures that changes that are local to the functionality or the user interface of services on a mobile device require only local changes to the vehicle's software.

SOLUTIONS

The main question for the implementation of an HMI integration is the level of abstraction at which the services interact with the HMI. We distinguish between the following five possibilities.

1. *Direct access* of the vehicle's I/O elements by the service. Here, the external service gets direct access to the I/O devices; for example, a display can switch between various services via a video switch or an LVDS multiplexer.

2. *Transmission of bitmaps.* Here, the service send bitmaps to the vehicle software; these bitmaps are then replicated on the vehicle's displays.

3. *Transmission of graphical primitives.* This variant is similar to the transmission of bitmaps. The difference is that images are not transmitted as complete bitmaps but are composed from graphical primitives. An example of a desktop remote HMI that follows this approach is X-Windows [3].

4. *Transmission of logical graphic primitives ("terminal mode").* This approach uses a logical protocol that controls abstract HMI objects. The vehicle software (on the head unit or another vehicle ECU) runs a presentation layer that provides abstract HMI objects for manipulation to the external services. Examples of such abstract HMI objects are menus, buttons, lists, list entries, string entry fields, and so on. Internally, the presentation layer knows how these abstract objects are presented to the user, but the interface between the presentation layer and the service contains no such information about the concrete representation. An interaction scenario proceeds as follows. The service transmits to the presentation layer the abstract HMI objects that are to be presented, along with their contents. When the user interacts with an abstract object (e.g., by selecting a menu item or pressing a button) the service performs the associated action and triggers a state transition of the execution logic. This state transition triggers modifications in the structure of abstract HMI objects, which, in turn, modifies the presentation layer.

5. *HMI generation for dynamic services.* This approach is similar to the "terminal mode" in that it separates the abstract HMI (menus, lists, buttons, and their execution logic) from the concrete presentation layer of the HMI. However, rather than calculating a concrete presentation at every interaction step,

the HMI generation approach generates a concrete HMI (including its graphical presentation layer and the concrete execution logic) only once for each dynamic service, namely at the time the service is registered into the system. This concrete HMI is generated based on a semantic description of the service and a description of its abstract HMI, using generation rules that are specific to the particular HMI concept implemented on the head unit.

The first three suggestions in this list are mentioned here for completeness only. They can be ruled out as implementation alternatives immediately because they make a comprehensive, uniform HMI impossible for several reasons. First, each device would have to have its own HMI programmed specifically for the particular configuration of I/O elements and against the particular interfaces provided by the vehicle. Second, the look and feel of the HMI would differ from device to device. Third, the execution logic would be distributed, local to each device. Fourth, service interoperability would be impossible, at least as far as the HMI is concerned. Fifth, priority management would have to take place at the device level. Sixth, the OEM would have no control over what gets presented to the user in which situation; as a result, it would be impossible for the OEM to ensure the various legal requirements.

Only the alternatives "terminal mode" and "HMI generation" allow the presentation layer to be logically decoupled from the services. BMW has implemented both of these solution alternatives prototypically.

RESULTS

1. The HMI of external services must be adapted to the specific I/O elements (iDrive commander, buttons, displays, etc.) in the vehicle.

2. The HMI of an external service must be integrated into a comprehensive, consistent HMI that covers all infotainment services that are available in the vehicle.

3. A seamless integration of the HMIs of various services is necessary in order to facilitate inter-service use cases. This requires that services interoperate at a high level of semantic abstraction.

4. For an integration of multiple services, a clear priority management is necessary at the granularity of actions, tasks and events.

Both the terminal mode approach and the HMI generation approach meet the important requirements discussed in this paper. The terminal mode amounts to a loose coupling that gives more power to the service side than the HMI generation approach. This property allows for a very flexible handling of situations that may not be foreseen at deploy time of the head unit. The terminal mode causes more communication between the service and the head unit than HMI generation; it makes error handling more difficult. The HMI generation approach has the advantage that the validity of the service descriptions (and therefore of the generated service HMI) can be checked entirely at HMI generation time. This is important property for error handling.

CONCLUSION

The flexible integration of CE devices into the automotive environment provides an opportunity to bring the latest infotainment functionality into the car quickly. However, if this integration is not done carefully, using the new services can become a nuisance to the customer rather than an added value.

The main challenge concerning the HMI integration of CE devices is the need to achieve a seamless integration of individual service HMIs into one comprehensive, intuitive HMI. Yet, from a technical perspective and from a process perspective, we aim for a decoupling of external services from the vehicle software.

In this talk we gave an overview of two solution paths: terminal mode and HMI generation, which elegantly achieve the technical decoupling and HMI integration.

REFERENCES

1. C. Grote, D. Weyl, R. Rau: „Integration mobiler Endgeräte — Problemstellungen und mögliche Lösungsszenarien." *Elektronik im Kraftfahrzeug.* VDI. Baden-Baden, October 2005.
2. A. Hildisch, J. Steurer, R. Stolle: „HMI generation for plug-in services from semantic descriptions." *Submitted.*
3. X Window System Technology, X.Org, www.x.org.