M. Koch, A. Butz & J. Schlichter (Hrsg.): Mensch und Computer 2014 Workshopband,
München: Oldenbourg Wissenschaftsverlag, 2014, S. 221-228.

Erste Erfahrungen aus der Entwicklung eines Android basierten Infotainmentsystems

Stefan Geisler[1], Benedikt Gorniak[2], Marc Jansen[3]

Institut Informatik, Hochschule Ruhr West[1]
Produktentwicklung Komponenten Interieur & Business-Cars, Brabus GmbH[2]
Research and Development, ToMM Apps UG (haftungsbeschränkt)[3]

Zusammenfassung

In den letzten Jahren ist die Verwendung mobiler Endgeräte im Automotive Bereich immer wichtiger geworden. Auf der einen Seite bringen immer mehr Personen ihre mobilen Geräte mit in ihr Auto und wollen hier auch auf verschiedene Funktionen des jeweiligen mobilen Geräts zugreifen können. Auf der anderen Seite haben sich mobile Geräte und die dort zum Einsatz kommenden Betriebssysteme aber auch als ideale Kandidaten für eine IT Unterstützung im Automotive Bereich herausgestellt. Das Ziel dieses Beitrages ist es, erste Erfahrungen aus der Entwicklung eines Infotainmentsystems auf Basis einer Android basierten Hardware vorzustellen.

1 Einleitung

Automobilhersteller stehen zunehmend vor der Herausforderung, insbesondere bei ihren Infotainmentsystemen mit dem rasanten Tempo aus dem Bereich der Smartphones und Tablets Schritt zu halten. Beiden Bereichen ist es gemein, dass die Nutzer mobil unterhalten werden sollen. So wachsen Anforderungen aus beiden Bereichen zusammen. Verstärkt wird dieser Effekt durch Cloud-Dienstleistungen (etwa zum Musik-Streaming) und soziale Netzwerke, die zunehmend Telefonate und SMS ersetzen. Die Nutzer wollen auch im Fahrzeug diese Dienste mit der gleichen Identität nutzen und idealerweise mit einem vergleichbaren Bedienkonzept, so dass kein ständiges Umdenken notwendig ist.

Die Schnelllebigkeit mobiler Dienste macht es erforderlich, dass in kurzen Zyklen neue Softwareversionen bereitgestellt werden. Da dies nicht zum traditionellen Entwicklungszeitraum eines Automobils passt, überlegen die Hersteller zunehmend Techniken zur App-Integration im Fahrzeug. Als Beispiel seien hier zwei Ansätze genannt: Der Hersteller Ford setzt auf eine offene API für sein SYNC-Radio-Navigationssystem, die es Drittanbietern erlaubt,

Apps für diesen Fahrzeugtyp zu entwickeln (Pulathaneli, 2014). Zum zweiten sei die von Google initiierte Open Automotive Alliance (Open Automotive Alliance, 2014) genannt, der sich Audi, GM, Honda, Hyundai und Nvidia angeschlossen haben. Audi hat direkt auf der CES2014 das Android basierte „Smart Display“ angekündigt.

Der Markt mobiler Endgeräte ist sehr heterogen, sowohl was Ausstattung, Formfaktor aber insbesondere auch das eingesetzte Betriebssystem angeht. Beispielhaft ist in Abbildung 1 die Verteilung mobiler Betriebssysteme (IDC, 2013) aus dem Jahr 2013 dargestellt.

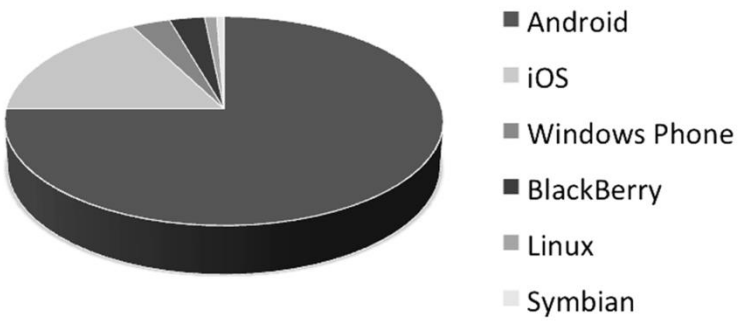

Abbildung 1: Verteilung mobiler Betriebssysteme im Jahr 2013, (IDC, 2013)

Auf der einen Seite bringt diese große Heterogenität für den Kunden/Anwender die Möglichkeit mit sich, dass Gerät seiner Wahl entsprechend gut auf seine Anforderungen anzupassen, auf der anderen Seite stellt es Software Entwickler vor die Herausforderung, die ein solch heterogener Markt mit sich bringt, wie z.B. das Problem der Auswahl der Zielplattform für die Entwicklung einer Software.

In dem hier konkret vorliegenden Fall, der Entwicklung eines Infotainmentsystems im Automotive Bereich, ist sehr frühzeitig die Entscheidung getroffen worden, dass wir uns für unsere Zielgruppe auf zwei mobile Betriebssysteme beschränken, nämlich Android und iOS, was auf der einen Seite einer theoretischen Marktabdeckung von ca. 92-93% entspricht, was bei näherer Betrachtung der anvisierten Zielgruppe eher noch etwas höher ausfallen dürfte.

Auf der anderen Seite erlaubte die hohe Heterogenität der verfügbaren mobilen Systeme eine Auswahl für das im Fahrzeug verbaute System, welche die unterschiedlichen Anforderungen in nahezu idealer Art und Weise mit einander verbindet. Hier fiel die Wahl auf ein Android basiertes System, da es aufgrund seiner der Offenheit seiner Schnittstellen und der Unterstützung von Schnittstellen anderer Systeme am flexibelsten erschien.

Im weiteren Verlauf beschreibt dieses Papier einige Erfahrungen, die wir während der Entwicklung des mobilen Infotainmentsystems gesammelt haben. Hierzu werden im folgenden Abschnitt zuerst die Anforderungen sowohl aus inhaltlicher als auch aus technischer Sicht dargestellt. Anschließend wird im dritten Abschnitt die Architektur des Systems dargestellt, welche es uns erlaubte, die formulierten Anforderungen flexibel umzusetzen. Der vierte Ab-

schnitt beschreibt dann unsere Erfahrungen aus der Entwicklung der Benutzerschnittstelle. Abschließend diskutiert der fünfte Abschnitt kurz die dargestellte Lösung und gibt einen Ausblick auf zukünftige Entwicklungen.

2 Anforderungen an das Infotainmentsystem

Wie bei jeder Neuentwicklung steht am Anfang die Definition hinreichender Anforderungen, die das spätere System erfüllen muss. Hierbei kann eine Unterteilung in inhaltliche und technische Systemanforderungen vorgenommen werden. Die inhaltlichen Anforderungen beschreiben den Funktionsumfang, insoweit er von dem Benutzer wahrgenommen wird. Die technischen Anforderungen beschreiben die Anforderungen, welche durch die Software erfüllt werden müssen, um für jeden Bediener in unterschiedlichen Fahrzeugen stets das gleiche Nutzungserlebnis sowie uneingeschränkte Funktionalität zu gewährleisten. Bei der folgenden Darstellung der inhaltlichen und technischen Anforderungen beschränken wir uns auf die Beschreibung der Anforderungen, die im Wesentlichen durch die in Abschnitt 3 dargestellte Architektur sowie der in Abschnitt 4 dargestellten Benutzerschnittstelle umgesetzt wurden.

2.1 Inhaltliche Anforderungen

Das Brabus-Infotainmentsystem unterscheidet sich in einem Punkt wesentlich von den bekannten Infotainmentsystemen (z.B. Mercedes COMAND): es ist in erster Linie nicht auf die Bedienung durch den Fahrer, sondern durch den Passagier ausgelegt. Hieraus ergibt sich ein geringerer Fokus auf das Bereitstellen klassischer Fahrzeuginformationen (z.B. Drehzahl oder Motortemperatur) bei stärkerem Fokus auf Internet- und Entertainmentapplikationen. Die Eignung des Systems während der Fahrt bedienbar zu sein ohne den Fahrer vom Verkehr abzulenken, steht damit ebenfalls weniger im Mittelpunkt.

Trotz dieser Unterschiede sollte die graphische Benutzeroberfläche sowohl in Hinblick auf das optische Erscheinungsbild, als auch in Hinblick auf das Nutzungserlebnis dem den Passagieren aus ihren anderen Fahrzeugen bekannten Mercedes-Benz COMAND ähneln und überdies die Brabus Markenidentität wiederspiegeln.

Da die mit dem Brabus-Infotainmentsystem ausgerüsteten Fahrzeuge weltweit im Einsatz sein werden, muss die Benutzeroberfläche unabhängig von der Sprache und dem kulturellen Hintergrund des Bedieners einfach verständlich und intuitiv bedienbar sein. Aus diesem Grund wurde festgelegt, die erste und zweite Ebene der Benutzeroberfläche ausschließlich graphisch ohne schriftliche Bezeichnung der Funktion auszuführen.

2.2 Technische Anforderungen

Die Business-Fahrzeuge, in denen das Infotainmentsystem Verwendung findet, sind modular aufgebaut. Abhängig von der gewählten Ausstattung sind also unterschiedliche Komponenten zu bedienen. Um diesem Umstand zu entsprechen, muss die Software flexibel in Hinblick

auf die jeweilige Fahrzeugausstattung sein. Ist das Fahrzeug beispielsweise mit Ambientebeleuchtung ausgestattet, so muss diese über das Menü bedienbar sein. Ist es das nicht, so darf der Punkt „Ambientebeleuchtung“ nicht Teil des Menüs sein.

Der Passagier eines Fahrzeuges ist es heute gewohnt, sein eigenes Smartphone oder Tablet mit ins Fahrzeug bringen zu können und vom Fahrzeug aus Zugriff auf die Daten des Gerätes zu haben. Diese Funktionalität sollte unabhängig vom Betriebssystem des jeweiligen Mobilgerätes auch im Brabus-Infotainmentsystem gewährleistet sein. Der Hauptfokus liegt hierbei auf der Wiedergabe von auf dem Mobilgerät gespeicherten Inhalten über die im Fahrzeug installierten Bildschirme sowie das Audiosystem.

3 Architektur des Gesamtsystems

Der Aufbau des hier beschriebenen Infotainmentsystems war mit verschiedenen Herausforderungen begleitet. Auf der einen Seite wurden durch die bereits beschriebenen Anforderungen einige der Herausforderungen aufgeworfen, auf der anderen Seite geschah die Entwicklung des Systems mit einem sehr engen Zeitplan.

Im ersten Teil dieses Abschnitts beschreiben wir den groben technischen Aufbau der Systemarchitektur. Durch diesen Aufbau ist eine der wesentlichen Anforderungen, die möglichst flexible Einbindung verschiedener Systeme von Drittanbietern, umgesetzt worden.

Um sowohl dem engen Zeitplan, als auch den technischen Herausforderungen gerecht zu werden sind bestimmte Vorkehrungen in Rahmen der Entwicklung der Softwarearchitektur vorgenommen worden, die im zweiten Teil dieses Abschnitts näher beleuchtet werden.

3.1 Technischer Aufbau

Die Grundidee des technischen Aufbaus war es, möglichst flexibel Geräte von Drittanbietern in die Systemarchitektur einbinden zu können. Auf Basis einer Analyse der zu berücksichtigenden Geräte wurden hierzu zwei Standardschnittstellen identifiziert, welche von den meisten der jeweiligen Geräte unterstützt werden. Auf der einen Seite erlauben die meisten modernen Geräte die Ausgabe ihres Bildes über eine standardisierte HDMI Schnittstelle, auf der anderen Seite erfolgt die Kommunikation häufig über Infrarot bzw. Wlan. Diese drei Schnittstellen haben sich als kleinster gemeinsamer Nenner zur Kommunikation mit verschiedenartigen Geräten heraus gestellt.

Um diese Schnittstellen geeignet umsetzen zu können wurde in der Gesamtarchitektur ein HDMI-Switch verplant, welcher zur Steuerung der visuellen Ausgaben der jeweiligen Geräte eingesetzt wird. Darüber hinaus wurde noch ein Gerät als Schnittstelle zur Infrarotkommunikation zu den entsprechenden Geräten mittels Wlan verbaut. Ein Nachteil der Infrarotsteuerung ist, dass sich ein Rückkanal in den seltensten Fällen, z.B. um dem Benutzer ein geeignetes Feedback zur Verfügung stellen zu können, realisieren lässt.

3.2 Softwarearchitektur

Die Hauptherausforderungen für die Softwarearchitektur ergaben sich im Wesentlichen aus zwei Anforderungen. Auf der einen Seite sollte während der Entwicklungsphase eine möglichst große Unabhängigkeit zwischen den an der Entwicklung der Kernkomponenten beteiligten Personen und den Entwicklern der Benutzerschnittstelle vorhanden sein. Diese Anforderung ging auf den sehr engen Zeitplan für die Entwicklung zurück, war aber auch dadurch getrieben, dass man zu einem späteren Zeitpunkt die Benutzerschnittstelle ggfs. vollkommen losgelöst von der Kernarchitektur verändern können wollte. Auf der anderen Seite sollte die Architektur der Gesamtsoftware eine möglichst flexible Adaption an unterschiedliche Konfigurationen im Fahrzeug ermöglichen, insbesondere der Ausstattung bzw. der Auswahl der unterschiedlichen Komponenten, die von dem zu entwickelnden Infotainmentsystem angesteuert werden sollten (z.B. Blu-ray Player, Apple TV, Ambilight) und sonstigen möglichen Schnittstellen, die sich z.B. daraus ergeben, dass Passagiere des Fahrzeugs ihre eigenen Geräte bzw. Datenquellen in das Fahrzeug bringen und dort anschließen, bzw. deren Daten konsumieren möchten.

Die möglichst große Unabhängigkeit zwischen den Entwicklern der Kernkomponenten und der Benutzerschnittstelle zu erreichen, erscheint auf den ersten Blick recht einfach umzusetzen, z.B. durch entsprechende Design Patterns (Gamma, et. al., 1994), die eine solche Trennung bereits vorsehen. Hier bietet sich im Android Umfeld vor allem das Model-View-Controller (MVC) (Reenskaug, 1979) Design Pattern an. Die hierdurch hervorgerufene Entkopplung zwischen der Funktionalität der einzelnen Komponenten und deren visueller Repräsentation erschien uns aber noch nicht weitreichend genug um eine vollkommen losgelöste Arbeit zwischen den Entwicklern der Kernkomponenten und der graphischen Benutzerschnittstelle zu ermöglichen. Aus diesem Grund haben wir eine Erweiterung des MVC Design Patterns, ähnlich wie in (Jansen & Markard, 2013) beschrieben implementiert. Die hieraus resultierende Architektur ist in Abbildung 2 dargestellt.

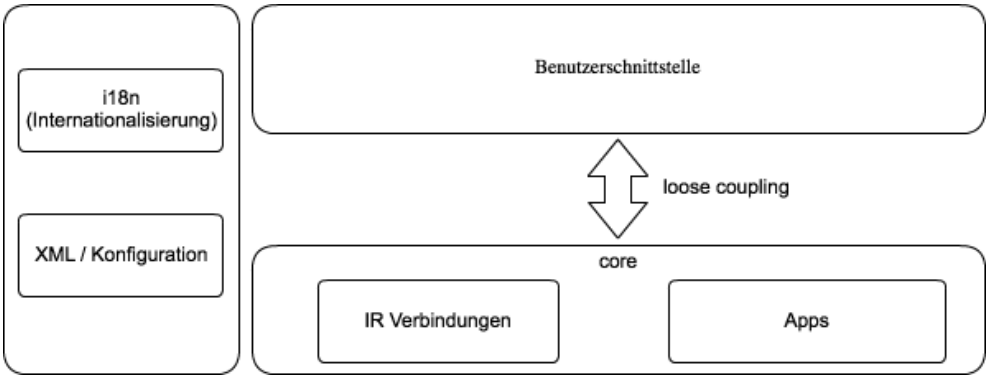

Abbildung 2: Darstellung der Softwarearchitektur zur losen Kopplung

Die hierdurch erreichte lose Kopplung zwischen der Benutzerschnittstelle und den Kernkomponenten, den Infrarot Verbindungen zu externen Geräten (z.B. Blu-ray Player, HDMI

Switch, ...) und den verschiedenen Android Apps (Skype, Ambilight Steuerung, ...) die im Fahrzeug verbaut worden sind, wird dabei auf Basis einer Konfigurationsdatei erzeugt. In dieser Konfigurationsdatei können sowohl für die einzelnen Komponenten der Benutzerschnittstelle entsprechende Ereignisse definiert werden, als auch für die jeweiligen Kernkomponenten. Die Vorgehensweise ist hierbei, analog zum Observer Design Pattern (Gamma, et. al., 1994) so, dass die Benutzerschnittstellen bei ihrer Aktivierung das ihnen zugeordnete Ereignis auslösen, welches wiederrum von der jeweiligen Kernkomponente empfangen wird woraufhin die entsprechenden Funktionalitäten ausgelöst werden.

Ebenfalls auf Basis dieser Konfigurationsdatei ist auch die zweite Anforderung der flexiblen Adaption des gesamten Infotainmentsystems an verschiedene Konfigurationen innerhalb des Fahrzeugs umgesetzt worden. Neben den Ereignissen, kann auf Basis der zentralen Konfigurationsdatei ebenfalls definiert werden, welche Komponenten überhaupt innerhalb des Fahrzeugs vorhanden sind. Des Weiteren kann flexibel definiert und konfiguriert werden, welche Benutzerschnittstellen innerhalb des Infotainmentsystems dargestellt werden sollen. Neben der reinen Darstellung der jeweiligen Benutzerschnittstellenkomponente kann auch noch deren Zugehörigkeit zu der nächst höheren Organisationseinheit (in der Regel auf Ebene des Karussells, s. hierzu Abschnitt 4), festgelegt werden.

4 Entwicklung des User Interfaces

Eine Grundprämisse bei der Entwicklung der Benutzerschnittstelle war eine Reduktion auf ein Minimum. Der Benutzer sollte sich im Wesentlichen in der Brabus-Welt bewegen, zugleich aber die Flexibilität des zugrundeliegenden Android-Systems nutzen können. Zugleich sollte die Identität der Firma Brabus transportiert werden ebenso eine Verwandtschaft zum vielen Kunden Vertrauten COMAND-System von Mercedes.

Eine Strukturierung der gewünschten Funktionen zeigte, dass zwei Menuebenen ausreichend sein würden. Die einzelnen Funktionen und Apps werden in fünf Kategorien eingeteilt, wobei jede Kategorie bis zu sieben Elemente beinhalten kann. Ein Element kann dabei eine eigene, integrierte Aktivität der Brabus-Applikation sein oder aber das Icon zum Aufruf einer fremden App (etwa Facebook oder Twitter).

Zur effizienten Bedienung werden sowohl die Auswahl der oberen Ebene als auch die Apps der aktuell ausgewählten Unterebene gleichzeitig angezeigt. Grafisch fiel die Wahl auf ein Karussell für Menuebene 1 und Icons im Smartphone Stil für Menuebene 2 (Abbildung 3). Das Karussell kann über den Touchscreen durch eine Wischbewegung gedreht werden bzw. durch die Auswahl einer einzelnen Kachel. Eine Verwandtschaft zum COMAND ist durch die Drehlogik gegeben, die im Original-Mercedes-System jedoch durch einen Dreh-Drücksteller bedient wird. Auch optisch ist sowohl eine Ähnlichkeit als auch eine Eigenständigkeit erkennbar. Die Brabus-Markenidentität wird durch Farbwahl und Verwendung des Logos transportiert. Die untere Leiste passt sich automatisch dem gewählten Eintrag der oberen Ebene an.

Der Touchscreen ist zwischen zwei Sitzen im Passagierraum im Landscape-Modus fest verbaut. Er ist das einzige Kontrollelement des Systems, nur ergänzt durch eine separat ausgeführte Lautstärkeregelung.

Abbildung 3: Hauptbildschirm des Media Center

Über ein Expertenmenu erhält der Nutzer zusätzlich die Möglichkeit, auf das Android-System zuzugreifen. So stehen erfahreneren Nutzern weitere Funktionen zur Verfügung bis hin zur Installation eigener Apps. Zurück gelangt der Nutzer über ein Brabus-Icon oder die Android Home-Taste.

Technisch ist das System als Launcher implementiert. Es wird automatisch bei Systemstart gestartet und kommt beim Druck auf die Android-Home-Taste in den Vordergrund. Somit ist es die zentrale Stelle für die Bedienung des Systems.

Aus Sicht der UI-Entwicklung hat sich die im vorangegangenen Abschnitt beschriebene Softwarearchitektur bewehrt. So konnte das UI in einer ersten Phase komplett parallel zum Anwendungskern entwickelt werden. Zudem ist ein Austausch von Grafiken, Icons und referenzierten Apps leicht durchführbar.

5 Diskussion und Ausblick

In diesem Papier haben wir unsere Erfahrungen aus der Entwicklung eines zum Einsatz in Fahrzeugen geeigneten Infotainmentsystems auf Basis einer Android basierten Plattform dargestellt. Zentral hierbei war ein wichtiger Unterschied zu bereits vergleichbaren und existierenden Plattformen, die im Wesentlichen auf eine möglichst wenig vom Fahrgeschehen ablenkende Bedienung durch den Fahrer optimiert sind. Bei unserem System war

von Anfang an eine Steuerung durch den Passagier des Fahrzeugs das Ziel der Gesamtentwicklung. Dies ist im Wesentlichen durch das Gesamtkonzept des Fahrzeugs begründet.

Die in diesem Papier beschriebene Architektur erlaubt auf der einen Seite eine flexible Umsetzung der dargestellten Anforderungen, bei gleichzeitiger Trennung zwischen verschiedenen Entwicklungsbereichen. So wurden in unserem Projekt z.B. die Entwicklung der Komponenten der Benutzerschnittstelle und die Entwicklung der Kernkomponenten des Systems, wie z.B. der Schnittstellen zu anderen Systemen, sowohl inhaltlich als auch technisch voneinander getrennt, um aus Zeitgründen eine parallele Bearbeitung ohne jeweilige Beeinflussung zu gewährleisten.

In Zukunft werden wir uns im Rahmen dieses Projekts mit der Umsetzung zusätzlicher Schnittstellen, z.B. zur weiteren Integration der vom Passagier ins Fahrzeug mitgebrachten Geräte kümmern. Darüber hinaus wird ein weiterer Punkt der zukünftigen Entwicklung die Einbindung neuer Schnittstellen zu bereits im Fahrzeug vorhandenen Systemen sein. Hierbei ist geplant sowohl Fahrzeug typische Schnittstellen (z.B. CAN-Bus) deutlich stärker in das Gesamtsystem zu integrieren, aber auch Schnittstellen zu zusätzlichen Fahrzeug untypischen Geräten weiter zu entwickeln.

Danksagung

Für die Erstellung des grafischen User Interface Designs und Mitarbeit beim Interaktionsdesign zeichnet Frau Tingting Ren verantwortlich (http://www.tingsdesign.de/).

Literaturverzeichnis

Gamma, R. , Helm, R., Johnson, R. E., Vlissides, J. (1994) *Elements of Reusable Object-Oriented Software*. Addison-Wesley

IDC Worldwide Quarterly Mobile Phone Tracker (May 2013)

Jansen, M., Markard, M. (2013) About an Extension of the Model-View-Controller Design Pattern for Increasing the Flexibility of Web Based Applications. In: *Proceedings of the 9th International Conference on Web Information Systems and Technology*

Open Automotive Alliance (2014). http://www.openautoalliance.net (letzter Zugriff: 9.6.2014)

Pulathaneli, T. (2014) Ein offener Ansatz zur App-Integration im Fahrzeug. In: ATZelektronik 9(1), Springer Automotive Media

Reenskaug, T. (1979) MVC: XEROX PARC 1978-79, *Xerox*

Kontaktinformationen

Prof. Dr. Stefan Geisler, E-Mail: stefan.geisler@hs-ruhrwest.de
Benedikt Gorniak, E-Mail: benedikt.gorniak@brabus.com
Prof. Dr. Marc Jansen, E-Mail: marc.jansen@tomm-apps.de