Eine abgesicherte Bedien- und Anzeigeschnittstelle zur Integration von Smartphone-Anwendungen in die Benutzeroberfläche von Fahrerinformationssystemen

Fabian Hüger

Konzernforschung Elektronik und Fahrzeug
Volkswagen AG
Brieffach 1777
38436 Wolfsburg
fabian.hueger@volkswagen.de

Abstract: Durch eine Bedien- und Anzeigeschnittstelle können externe Anwendungen (z.B. auf Smartphones) in die Benutzeroberfläche von Fahrerinformationssystemen integriert werden. Dabei wird die Benutzeroberflächenbeschreibung auf dem Server, von dem die Anwendung bezogen wird, signiert und im Fahrerinformationssystem auf Authentizität überprüft. Die Benutzeroberflächenbeschreibung definiert den gesamten Bedienablauf sowie die Layouts und schränkt somit die Änderungen durch die auf dem externen Gerät vorhandene Logik auf Inhaltsänderungen ein. Somit ist es nicht möglich, zuvor nicht zertifizierte Bedienabläufe im Fahrerinformationssystem zu realisieren. Im Vergleich zu anderen Lösungen wird dabei kein abgesicherter Bereich auf dem mobilen Gerät benötigt. Mit einer Entwicklungsumgebung können durch grafische Modellierung Benutzeroberflächen erzeugt werden, die über die abgesicherte Schnittstelle verfügbar sind.

1 Hintergrund

Ein vernetztes Fahrzeug bietet Zugang zu aktuellen Onlinediensten. Da die Lebenszyklen von Fahrzeugen und Onlinediensten sehr unterschiedlich sind, ist eine flexible Möglichkeit zur nachträglichen Integration von Anwendungen in Fahrerinformationssysteme erforderlich. Zur Integration der Anwendungen können diese in eine separate Laufzeitumgebung ausgelagert werden. Dadurch kann sichergestellt werden, dass die Grundfunktionalität des Fahrerinformationssystems nicht beeinflusst wird. Um Darstellung und Steuerung im Fahrerinformationssystem zu ermöglichen, werden Anzeige- und Bedieninformationen mit dem Fahrerinformationssystem ausgetauscht. Neben der Auslagerung auf ein separates Steuergerät liegt als Laufzeitumgebung aus Kostengründen die Verwendung eines beim Kunden eventuell schon vorhandenen Smartphones nahe. Dieses muss jedoch als unsicheres System angesehen werden. Da es mit bestehenden Technologien über eine externe Schnittstelle

mit dem Fahrerinformationssystem kommuniziert, können bei fehlender Absicherung unerwünschte Anwendungen im Fahrerinformationssystem realisiert werden. Unerwünschte Anwendungen können den Fahrer ablenken, die Geschäftsmodelle des Herstellers (OEM) gefährden oder Imageschaden verursachen. Zum Beispiel sind Bewegtbilder oder zeitlich abhängige Bedienvorgänge durch die Richtlinien zur Fahrerablenkung untersagt [Eur06]. So ist zum Beispiel die Video/TV-Darstellung in Fahrerinformationssystemen blockiert. Geschäftsmodelle können dann gefährdet werden, wenn Funktionalitäten, die vom OEM als Mehrausstattung verkauft werden, durch Dritte realisiert werden können und somit der Kaufanreiz der Mehrausstattung verkleinert wird. Eine Mehrausstattung könnte zum Beispiel eine Navigationsfunktion sein. Imageschaden kann bei instabilen Anwendungen oder durch Schadsoftware erzeugt werden.

2 Stand der Technik

In der Konsumerelektronik existieren Methoden um nur bestimmte Anwendungen auf Geräten realisieren zu können. Als Beispiel kann das Apple iPhone genannt werden. Durch eine Rechteüberprüfung des Betriebssystems können nur Anwendungen ausgeführt werden, die aus dem Apple App-Store bezogen worden sind. Durch eine Software-Änderung am Gerät („jailbreak“) können jedoch auch andere Anwendungen ausgeführt werden.

Im automobilen Kontext wird zurzeit Mirror Link [Car12] entwickelt[1]. Dabei wird der Bildschirminhalt eines Smartphones zum Fahrerinformationssystem übertragen und dort dargestellt. Eingaben im Fahrerinformationssystem werden an das Mobilgerät weitergeleitet. Dabei wird das VNC-Protokoll verwendet. Um nur bestimmte Anwendungen zuzulassen, ist jeder Anwendung ein Attribut zugeordnet, das die Anwendung klassifiziert. Das Attribut wird im Fahrerinformationssystem überprüft und die Darstellung entsprechend zugelassen oder verhindert. Der Mechanismus, der bestätigt, dass das mobile Gerät nur zertifizierte Anwendungen ausführt, basiert auf X.509-Zertifikaten [IET08] und standardisierten Mechanismen der Trusted Computing Group [Tru12]. Kostiainen et al. beschreiben den Mechanismus im Detail [KAE11]. Dabei soll der private Schlüssel des Mobilgerätes in einem abgesicherten Speicherbereich abgelegt werden und die Anwendung in einer abgesicherten Laufzeitumgebung ausgeführt werden. In vielen aktuellen Mobilgeräten ist weder eine abgesicherte Laufzeitumgebung noch ein abgesicherter Speicherbereich vorhanden.

Wenn der Schlüssel nicht in einem abgesicherten Bereich abgelegt wird bzw. keine abgesicherte Laufzeitumgebung vorhanden ist, ist ein Man-in-the-Middle-Angriff möglich, bei dem eine schadhafte Anwendung sich als eine valide andere ausgeben kann und somit unerwünschten Anwendungen die Darstellung im Fahrerinformationssystem ermöglicht.

[1] Der vom Car Connectivity Consortium entwickelte Standard „Mirror Link“ wurde zunächst unter dem Namen „Terminal Mode“ entwickelt.

Ziel ist also sicherzustellen, dass nur bestimmte Anwendungen Zugriff auf die Benutzeroberfläche (User Interface – UI) haben und diese nur so anpassen dürfen, dass die Richtlinien zur Fahrerablenkung eingehalten werden und keine neuen Anwendungen möglich sind. Dabei soll kein gesicherter Bereich auf dem mobilen Gerät erforderlich sein.

In [Hüg11] haben wir einen Ansatz vorgestellt, bei dem die externe Anwendung (z.B. auf einem Smartphone) ihre Benutzeroberfläche in Form von Benutzeroberflächenbeschreibungen kommuniziert. Abbildung 1 verdeutlicht den Ansatz. Auf der Anwendungsplattform wird neben der Logik die Benutzeroberfläche von einem Remote-UI Service über einen http-Server dem Fahrerinformationssystem zur Verfügung gestellt. Der Programmablauf wird durch eine Abfolge unterschiedlicher vorgegebener Views definiert. Die Views können mit vorgegebenen View-Elementen befüllt werden. Im Fahrerinformationssystem werden die Benutzeroberflächenbeschreibungen interpretiert und den Views bzw. View-Elementen vorgegebene Layouts zugeordnet. Die Layouts sind neben der Benutzeroberfläche des Gesamtsystems fest in der Main-Unit vorhanden und umfangreich getestet, um die Testanforderungen eines Seriensystems zu erfüllen. Zusätzlich kann durch die festen Layouts sichergestellt werden, dass die Anforderungen an die Benutzeroberfläche hinsichtlich der Fahrerablenkung erfüllt werden. Eine Beschränkung auf bestimmte Anwendungen ist jedoch bisher nicht vorgesehen. Im Folgenden wird basierend auf diesem Ansatz eine Möglichkeit der Absicherung erläutert, bei der kein abgesicherter Speicherbereich erforderlich ist.

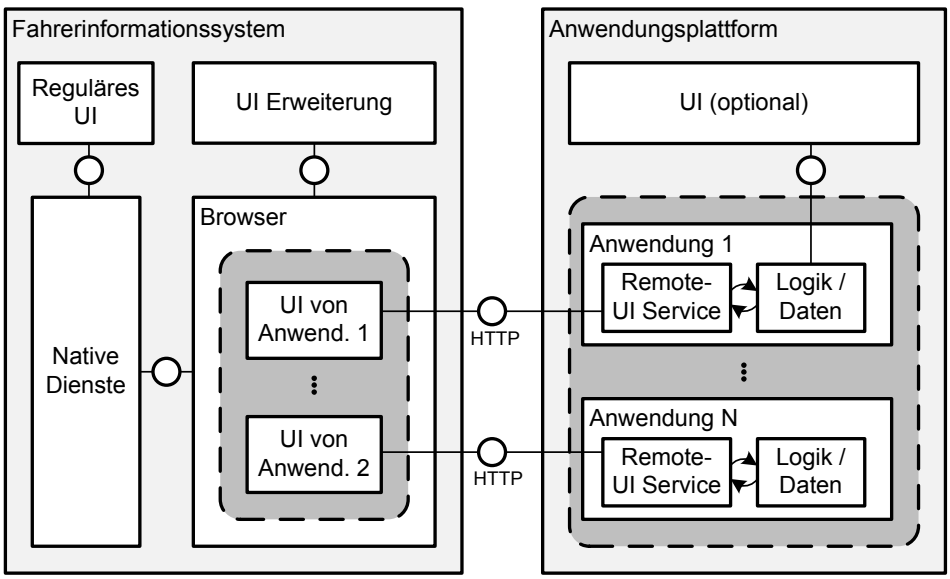

Abbildung 1: Integration der externen Anwendungen in das Fahrerinformationssystem

3 Abgesicherte Schnittstelle zur Erweiterung von Fahrerinformationssystemen

Zur Absicherung der Schnittstelle aus [Hüg11] verwenden wir die Eigenschaft, dass die Benutzeroberfläche in Form von Benutzeroberflächenbeschreibungen (View-Typen und View-Elemente) übertragen wird. Dabei wird die Benutzeroberfläche in feste Anteile (UI) und veränderliche Anteile aufgeteilt (ΔUI). Die festen Anteile beinhalten den Ablauf der Bedienung und die Layouts. Die veränderlichen Anteile sind die Inhalte, die von der Anwendung erzeugt und im Rahmen der festen Layouts und Abläufe dargestellt werden. Die Absicherung basiert auf einer Zertifizierung der festen UI-Anteile auf dem Server und der Überprüfung im Fahrerinformationssystem. Abbildung 2 verdeutlicht den Ablauf der Zertifizierung und Überprüfung. Der Server stellt nach X.509-Standard [IET08] ein Zertifikat (Cert) für das UI aus und überträgt es mit der Anwendung an das externe Gerät. Das Zertifikat enthält zusätzlich zu den Standardfeldern einen Hashwert des UI. Das externe Gerät leitet das UI und das Zertifikat an das Fahrerinformationssystem weiter. Die Applikationslogik wirkt sich nur in einer Änderung des UI in Form von Inhalten aus (ΔUI). Das UI wird nur dann auf dem Fahrerinformationssystem berücksichtigt, wenn das zugehörige Zertifikat erfolgreich überprüft wurde und der darin enthaltene Hashwert des UI mit dem Hashwert des mitgelieferten UI korrespondiert. Der Schlüssel wird bereits bei Auslieferung im Fahrerinformationssystem hinterlegt. Anwendungen mit verändertem bzw. nicht-zertifiziertem UI bleiben unberücksichtigt. Die Gefahr durch einen Man-in-the-Middle-Angriff beschränkt sich somit auf Änderungen innerhalb der fest vorgegebenen Bereiche. Der Titel der Anwendung und der Ablauf z.B. von einer Liste mit vorgegebenen Einträgen, über einen Speller zu einem variablen Text können zum Beispiel fest definiert sein. Innerhalb dieses Rahmens ist es schwierig, neue unerwünschte Anwendungen zu realisieren.

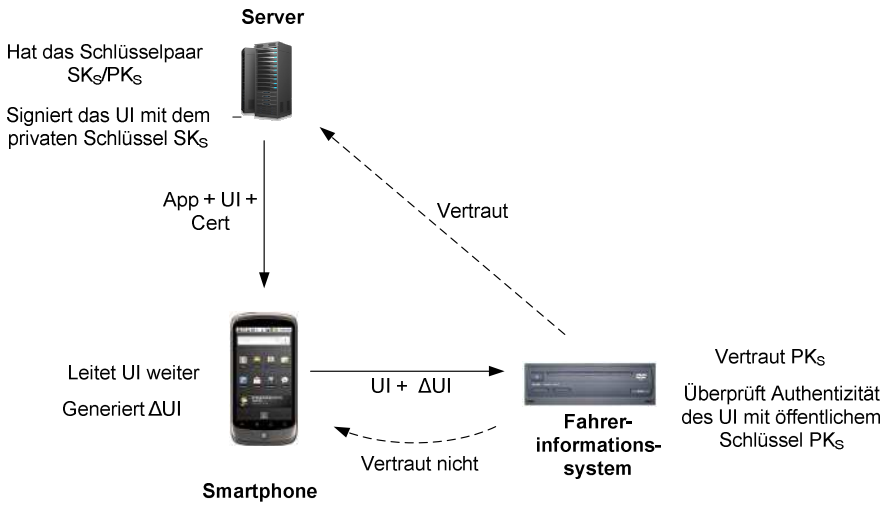

Abbildung 2: Zertifizierung und Überprüfung der Benutzeroberflächenbeschreibungen

4 Werkzeugunterstützung

Im Gegensatz zu mit bekannten Entwicklungsumgebungen (SDK) oder Sprachen erzeugten Benutzeroberflächen (z.B. Android oder HTML5) stehen bei der Entwicklung von Anwendungen für die benannte Schnittstelle nur bestimmte vorgegebene Layouts für die Benutzeroberfläche zur Verfügung. Zur Unterstützung bei der Erstellung von kompatiblen Anwendungen haben wir eine Entwicklungsumgebung in Form eines Eclipse Plug-Ins erstellt, das auf dem Eclipse Graphical Modeling Framework [The12] basiert. Damit lassen sich Bedienabläufe grafisch modellieren, sodass auch mit dem speziellen SDK unerfahrene Entwickler komfortabel Anwendungen mit einer Benutzeroberfläche für Fahrerinformationssysteme erstellen können. Dabei können auch bestehende Android-Anwendungen um eine Benutzeroberfläche für Fahrerinformationssysteme erweitert werden. Eigenschaften der Views, wie zum Beispiel Titel oder Icons, werden über Properties definiert. Abbildung 3 zeigt die Modellierung einer Benutzeroberfläche mit dem Plug-In. Aus dem Modell wird Code generiert, in dem bereits Klassen für Methodenaufrufe vorgesehen sind. Zusätzlich werden bei der Codegenerierung Libraries eingebunden, die für die Kommunikation mit dem Fahrerinformationssystem zuständig sind. Zum Testen der Benutzeroberfläche kann, auch ohne Deployment auf das mobile Gerät, ein http-Server mit der Benutzeroberflächenbeschreibung gestartet werden und mittels Simulation des Fahrerinformationssystems ausgeführt werden.

In einem Codecamp mit Studenten wurde das SDK erprobt und festgestellt, dass durch das Plug-In eine schnelle Einarbeitung in die Entwicklung für Fahrerinformations-systeme möglich ist.

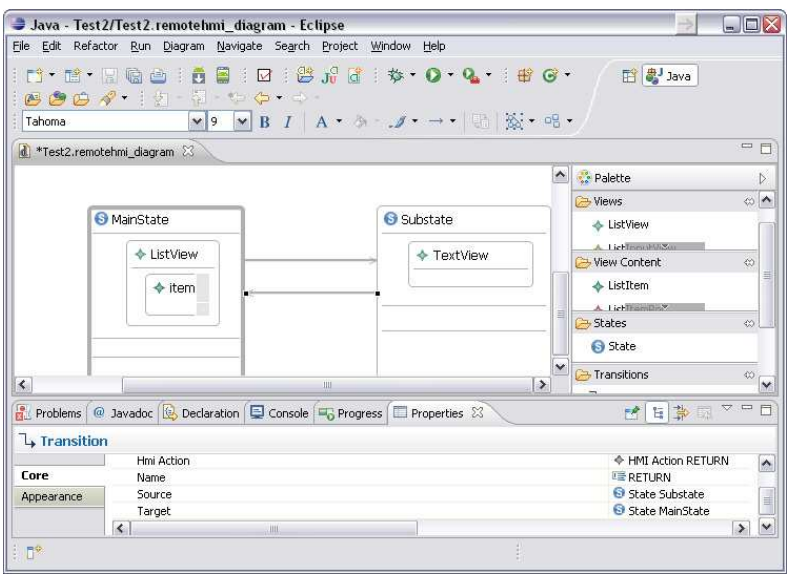

Abbildung 3. Plug-In für Eclipse zur Erzeugung von Benutzeroberflächen für Fahrerinformationssysteme

5 Zusammenfassung

Der beschriebene Ansatz zur Integration von Smartphone-Anwendungen in die Benutzeroberfläche von Fahrerinformationssystemen eignet sich um lediglich bestimmte Anwendungen zu signieren und zuzulassen. Durch die Trennung zwischen festen und veränderlichen Benutzeroberflächenanteilen können durch Man-in-the-Middle-Angriffe nur veränderte Inhalte innerhalb vorgegebener Layouts realisiert werden. Die Erstellung und Darstellung neuer Anwendungen (z.B. mit einem neuen Titel) ist nicht möglich. Die Signierung und Überprüfung wird auf dem Server bzw. auf dem Fahrerinformationssystem durchgeführt, und nicht auf dem unsicheren Smartphone. Somit ist kein gesicherter Speicherbereich auf dem Smartphone erforderlich. Mittels Eclipse Plug-In können durch grafische Modellierung Benutzeroberflächen erzeugt werden, die über die abgesicherte Schnittstelle verfügbar sind.

Literaturverzeichnis

[Car12] Car Connectivity Consortium. http://www.mirrorlink.com/, 2012.

[Eur06] European Union. Recommendations on safe and efficient in-vehicle information and communication systems: update of the European Statement of Principles on human machine interface, December 2006.

[Hüg11] Fabian Hüger. User interface transfer for driver information systems: a survey and an improved approach. In *AutomotiveUI'11, November 29-December 2, 2011, Salzburg, Austria*, 2011.

[IET08] IETF. RFC 5280, Internet X.509 Public Key Infrastructure Certificate, May 2008.

[KAE11] Kari Kostiainen, N. Asokan, and Jan-Erik Ekberg. Practical property-based attestation on mobile devices. In Jonathan McCune, Boris Balacheff, Adrian Perrig, Ahmad-Reza Sadeghi, Angela Sasse, and Yolanta Beres, editors, *Trust and Trustworthy Computing*, volume 6740 of *Lecture Notes in Computer Science*, pages 78–92. Springer Berlin / Heidelberg, 2011.

[The12] The Eclipse Foundation. Graphical Modeling Project (GMP) [Online]. Available: http://www.eclipse.org/modeling/gmp/, 2012.

[Tru12] Trusted Computing Group. http://www.trustedcomputinggroup.org/, 2012.