Fahrsituationsspezifische Datenverteilung im Verteilten Umgebungsmodell für Fahrzeugsoftware

Andreas Hermann
Institut für Angewandte Forschung
Fachhochschule Ingolstadt
85049 Ingolstadt

Abstract: Durch die stetig wachsende Zahl von Sensoren im Automobil nimmt das Datenvolumen auf dem Bussystem rapide zu. Daher stellt das Verteilte Umgebungsmodell für Fahrzeugsoftware (VerUM) Mechanismen zur Verfügung um den Datenfluss im Automobil an die gegenwärtige Fahrsituation anzupassen. Ferner werden Konzepte vorgestellt, die eine ortstransparente, ereignisgesteuerte Datenbereitstellung für Applikationen ermöglichen. Dies gestattet es den Entwicklern von Applikationen sich auf die Kern-Algorithmik zu fokussieren. Datenbereitstellung sowie Fahrsituationsbeschreibung und -analyse werden durch VerUM geleistet.

1 Einleitung

Durch die stetig wachsende Zahl von Sensoren im Automobil nimmt das Datenvolumen auf dem Bussystem rapide zu. Ferner werden immer höhere Anforderungen an moderne Komfort- und Sicherheitsapplikationen gestellt. Ein Beispiel hierfür sind Adaptive Cruise Control (ACC) Systeme, welche riskante Situationen wie einscherende Fahrzeuge [DBSS04] berücksichtigen.
Um eine Plattform für zukünftige Automotive-Applikationen zu schaffen haben sich führende Automobilhersteller und Zulieferer im AUTOSAR-Konsortium [AUT06] zusammengeschlossen. Hierbei wurden generelle Probleme wie Hardwareabstraktion, echtzeitfähige Kommunikation und Standardisierung von Basissoftware gelöst. Allerdings werden entscheidende Bereiche, wie die Sensordatenverteilung oder die Analyse von Fahrsituationen, nicht durch AUTOSAR abgedeckt. Aus diesem Grund wurde das Verteilte Umgebungsmodell für Fahrzeugsoftware (VerUM) entwickelt. VerUM bietet ein Situationsmodell (SM) und eine Situationsanalyse (SA). Hierdurch wird ein fahrsituationssensitives System geschaffen. Auf Basis dieser Fahrsituationen wird die Verteilung von Sensordaten angepasst. Um die Datenakquisition von entfernten Steuergeräten zu beschleunigen, wird eine proaktive Datenverteilung für prognostizierte Fahrsituationen vorgeschlagen.

Dieser Artikel ist folgendermaßen gegliedert. In Abschnitt 2 wird die Architektur von VerUM vorgestellt, die fahrsituationsspezifische Datenverteilung folgt in Abschnitt 3. Erste Evaluierungsergebnisse werden in Abschnitt 4 vorgestellt. Der Artikel wird in Abschnitt 5 zusammengefasst.

2 Verteiltes Umgebungsmodell für Fahrzeugsoftware (VerUM)

Embedded Middleware ist häufig auf Object Request Brokern (ORB) aufgebaut, wie beispielsweise RT CORBA [Obj05] und TAO [SLM98]. ORB Architekturen sind auf das Ausführen von entfernten Funktionen optimiert. Dies deckt sich nicht mit den Anforderungen im Automobil, da aufgrund der harten Echtzeitbedingungen die Ausführung von Funktionen in der Regel lokal erfolgt. Vielmehr steht der Datenfluss von der Sensorik zur Algorithmik im Vordergrund. VerUM begegnet dieser Anforderung mit einer mehrschichtigen Architektur (siehe Abbildung 1) für Sensordatenfusion und -verteilung. Über das

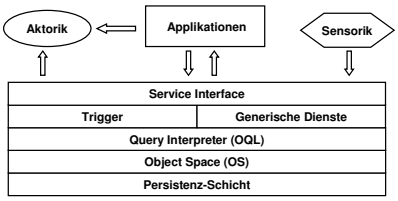


Abbildung 1: Schichtenmodell einer VerUM Instanz

Service Interface können Applikation auf die Funktionalität von VerUM zugreifen. Trigger sind eventgesteuerte Programmiermechanismen zur Datenakquisition und -manipulation. Über Generische Dienste werden grundlegende Funktionalitäten wie die fahrsituationsspezifische Datenverteilung und das ereignisgesteuerte Programmiermodell bereitgestellt. Der Object Space ist ein systemweiter Container für Sensordaten. Die im Object Space gespeicherten Daten sind für einen ortstransparenten Zugriff über den Query Interpreter indiziert. Die Persistenz-Schicht bietet grundlegende Funktionen für das persistente Speichern von Daten und Systemzuständen.

Um eine flexible und komfortable Datenbereitstellung zu gewährleisten, verfügt VerUM über ein ereignisgesteuertes Programmiermodell (EGP). Das EGP ist die Implementierung des Beobachter-Musters [GHJV04] im Kontext eines verteilten Systems. Alle Datenkonsumenten übermitteln ihre Registrierungen $b = \langle d, e \rangle$, mit d als eindeutige ID eines Datentyps und e einem Ereignis auf d, an eine VerUM Instanz. Folglich beschreibt die Menge B^g mit $b_i^g \in B^g$ alle i Registrierungen eines Datenkonsumenten g. Datenproduzenten registrieren eine Menge O von Datentypen, welche sie zur Laufzeit erzeugen.
Während der Initialisierungsphase wird das Set B^g an alle angeschlossenen VerUM Instanzen verteilt. Diese untersuchen B^g, ob Ereignisse, welche die Registrierungen in B^g erfüllen, auf dem lokalen Steuergerät auftreten. Übereinstimmungen werden in der lokalen Verteilungstabelle T gespeichert und führen zu einer Benachrichtigung von g, wenn das Ereignis für b_i^g zur Laufzeit auftritt. Dieser Mechanismus ist ortstransparent und benachrichtigt Datenkonsumenten unabhängig von ihrer Lokalität.

Des weiteren stellt VerUM ein einheitliches Datenmodell für Sensordaten sowie Algorithmen zur Vorverarbeitung von diesen, zur Verfügung. Wie Abbildung 2 zeigt werden Sensordaten über Filter-, Assoziations-, Fusions- und Klassifizierungsalgorithmen zu ei-

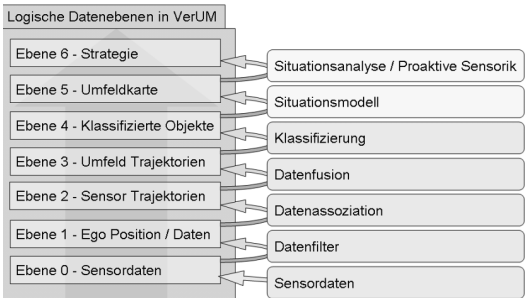


Abbildung 2: Datenrepräsentationsebenen in VerUM

ner einheitlichen Darstellung der Umgebung um das eigene Fahrzeug zusammengeführt. Dies geschieht im Situationsmodell (SM) auf Ebene 5. Basierend auf diesem SM wird eine Situationsanalyse (SA) durchgeführt, welche das Verhalten des eigenen Fahrzeugs und die Beziehungen zu anderen Verkehrsteilnehmern analysiert. Die Konzepte für SA und SM sind in [HL07] näher erläutert. Ein vergleichbarer Ansatz zur Sensordatenvorverarbeitung wurde von Wender et al. in [WWFD06] vorgestellt. Allerdings steht hierbei die Sensordatenfusion im Mittelpunkt und Bereiche wie SM und SA werden nicht näher betrachtet.
Durch diese Art der Datenrepräsentation abstrahiert VerUM von der zur Verfügung stehenden Sensorik. Sensordaten werden in VerUM-Objekte überführt und die Algorithmik der Ebenen 1-4 somit von spezifischen Sensordaten losgelöst. Hierdurch wird eine generische Sensordatenverarbeitung erreicht. Qualitativ höherwertige Sensoren (z. B. Laserscanner, Kamera) verbessern zwar die Ergebnisse von SM und SA, sind für die grundlegende Funktion von VerUM allerdings nicht zwingend erforderlich.

3 Fahrsituationsspezifische Datenverteilung (FSD)

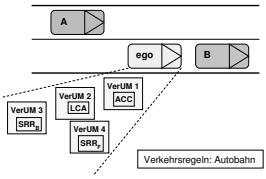
Ein bedeutendes Ziel des Automotive Software Engineering ist es die Buslast zu reduzieren. Aus diesem Grund sieht VerUM eine FSD zur Reduzierung des Datenvolumens auf dem Kommunikationsmedium vor. Um eine FSD zu ermöglichen, muss das Konzept der EGP um Fahrsituationen erweitert werden. Nach [HL07] kann eine Fahrsituation durch den 3-Tupel $\langle Verkehrsregeln, Aktion, Interaktion\rangle$ beschrieben werden. Das EGP kann nun um die Komponente Fahrsituation erweitert werden, indem $B^{g\prime}$ mit $b_i^{g\prime} \in B^{g\prime}$ und $b' = \langle d, e, s\rangle$ alle i Registrierungen eines Datenkonsumenten g in der Fahrsituation s beschreibt. Anstatt B^g wird nun $B^{g\prime}$ in der Initialisierungsphase verteilt. Die Benachrichtigung von g erfolgt analog zum Konzept der EGP aus Abschnitt 2.
Mit FSD kann die Datenmenge, welche zu einem Datenkonsumenten g transferiert werden muss, signifikant reduziert werden. Sei h die Menge an Daten die g in kürzeren Zyklen empfängt als die Fahrsituationen wechseln, so entsteht ein kontinuierlicher Datenstrom zu g. In einem System N, das Daten nicht fahrsituationsspezifisch verteilt, kann die Menge der an g übertragenen Daten, über die Zeitspanne $t_{pN} = t$, mit t als Uptime von N, durch

$\int_0^t h\ dt_{pN} \approx h \cdot t$ ausgedrückt werden.
In einem System D mit FSD und Uptime t ist die Menge der an g übertragenen Daten definiert als $\int h\ dt_{pD}$, mit $t_{pD} = \sum_{i=0}^{n} t_{p_i D}$, als der Zeitspanne die eine Fahrsituation i in t aktiv ist und n als die Anzahl der aktiven Fahrsituationen von g in t. Somit kann die an g in t übertragene Datenmenge mit $h \cdot t_{pD}$ angenähert werden. Aus $t_{pN} = t$ und $t_{pD} \leq t$ folgt somit $t_{pN} \geq t_{pD}$. In einem normalen Verkehrsszenario mit wechselnden Fahrsituation wird somit in D eine geringere Datenmenge an g übertragen als in N. Daher kann durch FSD die Buslast im Automobil gesenkt und freie Ressourcen für weiterführende Applikationen geschaffen werden.

Die in VerUM vorgestellte SA trifft Entscheidungen über die momentane Fahrsituation auf der Basis von Wahrscheinlichkeiten. Eine Verschiebung der Wahrscheinlichkeiten von einer Fahrsituation auf eine Andere deutet auf einen Wechsel der Fahrsituation hin. Somit kann eine zukünftige Fahrsituation mit einer bestimmten Wahrscheinlichkeit prognostiziert werden. Diese Information nutzt VerUM um eine proaktive Verteilung von Daten anzustoßen, welche in der prognostizierten Fahrsituation benötigt werden. Sei s die momentan gültige Fahrsituation, H ein VerUM das in s die Verteilungstabelle T_s bearbeitet und s' die durch SA prognostizierte nächste Fahrsituation, so berechnet H die proaktive Verteilungstabelle $T'_{s'} := \{b_i | b_i \in T_{s'} \wedge b_i \notin T_s\}$. Nach Abschluss dieser Berechnung wird $T'_{s'}$ im Hintergrund verarbeitet. So werden die Echtzeitanforderungen der momentanen Fahrsituation nicht beeinträchtigt. Registrierungen in $T'_{s'}$ unterscheiden sich von Registrierungen in T_s. Insofern die Aktualität der Daten gewährleistet werden kann, verteilt VerUM die, diesen Registrierungen zugeordneten Daten, proaktiv in den Zyklen vor dem Situationswechsel. Allerdings werden Applikationen nicht über proaktiv verteilte Daten benachrichtigt und somit auch nicht in ihrer Arbeit in s beeinträchtigt. Nach einem Situationswechsel auf s' stehen Eingabedaten bereits lokal auf den Steuergeräten der Applikationen zur Verfügung. Dies reduziert die Latenzzeiten bei einem Wechsel der Fahrsituation signifikant. Ferner werden Netzwerk- und Hardwareressourcen während der rechenintensiven Phase des Situationswechsels geschont.

4 Experimentelle Ergebnisse

In Abbildung 3 ist der Versuchsaufbau für die Evaluierung der FSD dargestellt. Das Ego-Fahrzeug verfügt über vier Steuergeräte mit VerUM Instanzen. Diese VerUMs kontrollieren das ACC, den Spurwechselassistent und konvertieren die aufgenommenen Daten der Radare (SRR_F und SRR_B) in VerUM Objekte. Bezugnehmend auf die Definition einer Fahrsituation aus [HL07] sind in der Tabelle in Abbildung 3 typische Verkerssituationen auf einer Autobahn dargestellt. Diese wurden im Evaluierungsszenario durchlaufen und das bewegte Datenvolumen zwischen den VerUM Instanzen aufgezeichnet. Wie in Abbildung 4 zu sehen ist, reduziert FSD das zu übertragende Datenvolumen signifikant. Dies bestätigt die in Abschnitt 3 aufgestellten Gleichungen und zeigt die Fähigkeit von FSD zur signifikanten Reduzierung des zu übertragenden Datenvolumens.



Aktionen	Interaktionen	ID
Straße folgen	Keine Interaktionen	(1)
Straße folgen	Annähern	(2)
Spurwechsel	Passieren	(3)
Straße folgen	Passieren	(4)
Spurwechsel	Keine Interaktionen	(5)
Straße folgen	Keine Interaktionen	(6)

Aktionen	Interaktionen	ID
Straße folgen	Annähern	(7)
Spurwechsel	Annähern & Passieren	(8)
Straße folgen	Annähern & Passieren	(9)
Straße folgen	Passieren	(10)
Straße folgen	Keine Interaktionen	(11)
Spurwechsel	Keine Interaktionen	(12)

Abbildung 3: Versuchsaufbau für FSD

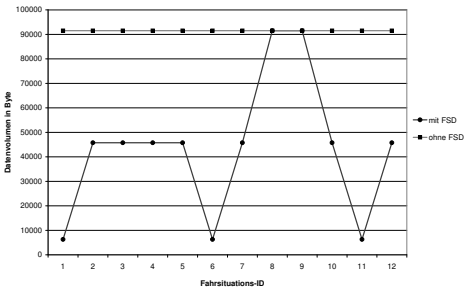


Abbildung 4: Datenvolumen im Versuchszenario

5 Zusammenfassung

Dieser Artikel befasst sich mit der Problematik der Datenverteilung im Automobil. Es wird VerUM vorgestellt, welches mit einem EGP und einer ortstransparenten, fahrsituationsspezifischen Datenverteilung, den Anforderungen von Applikationen im Automobil gerecht wird. Ausgehend von einer eindeutigen Fahrsituation wird eine FSD vorgeschlagen. Diese hat in ersten experimentellen Versuchsaufbauten das Potenzial zur signifikanten Senkung des Datenvolumens auf dem Bussystem gezeigt. Die in diesem Artikel beschriebenen Konzepte werden auf einem Versuchsfahrzeug der AUDI AG weiter evaluiert. Die bisherigen Testfahrten bestätigen das in Abschnitt 4 dargestellte Versuchsergebnis.

Literatur

[AUT06] AUTOSAR GbR. *AUTOSAR Technical Overview V 2.0.1*, June 2006.

[DBSS04] Ismail Dagli, Gabi Breuel, Helmut Schittenhelm und Alexander Schanz. Cutting-in vehicle recognition for ACC systems- towards feasible situation analysis methodologies. In *Proceedings of the 2004 IEEE Intelligent Vehicles Symposium*, 2004.

[GHJV04] Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides. *Entwurfsmuster*. Addison-Wesley, 2004.

[HL07] Andreas Hermann und Stefan Lutz. Situation based Data Distribution in a Distributed Environment Model. In *Proceedings of the 2007 IEEE Intelligent Vehicle Symposium*, 2007.

[Obj05] Object Management Group. Real-Time CORBA Specification Version 1.2, 2005.

[SLM98] Dougals C. Schmidt, David L. Levine und Sumedh Mungee. The Design of the TAO Real-Time Object Request Broker. *Computer Communication*, 21(4), April 1998.

[WWFD06] Stefan Wender, Thorsten Weis, Kay Fuerstenberg und Klaus Christian Juergen Dietmayer. Feature Level Fusion for Object Classification. *PReVent ProFusion e-Journal*, 1:31–36, 2006.