ADTF

Video

Entwicklung FAS

Flexible Entwicklungsplatform

Synchron Fahrzeugdaten

Weiterverarbeitung

Visualisierung

Reaktion

Funktionseigenentwicklung

VW Konzern Konzept bis Serie

Autonomes Fahren, Lichttecknik, Algorithmusentwicklung SerienFAS

Evaluierung von Sensorik

Zeitsynchron, dynamische Konfiguration, Wiederverwendbarkeit, gleiches Tool Konzept bis Serienentwicklung 🡪 Zeit, und Kosteneinsparung

**ADTF Software-Umgebung für Applikationen und Tools**

ADTF (**A**utomotive **D**ata and **T**ime-**T**riggered **F**ramework) ist eine Software-Umgebung, in der neue Funktionalitäten für Fahrerassistenz- oder Fahrzeugsicherheitssysteme einfach erstellt werden können. Ermöglicht wird dies durch eine Vielzahl an einsatzbereiten Modulen und durch eine umfangreiche Sammlung von Programmen und Dokumentationen. ADTF wird durch die Audi Electronics Venture GmbH seit 2001 in Zusammenarbeit mit verschiedenen Audi Fachabteilungen entwickelt. Die Software wird seit Januar 2008 durch die EB (Elektrobit) vertrieben. EB verkauft Lizenzen auch an andere Automobilhersteller sowie an Automobil-Zulieferer.

ADTF basiert auf einer durch Schnittstellen offen gelegten Gesamtarchitektur. Viele der nicht-funktionalen Eigenschaften des ADTF liegen diesem Schnittstellenkonzept zugrunde. Der ADTF-Kern definiert ein Software-Framework für Applikationen und Tools im Rahmen der Entwicklung von elektronischen Systemen. Ziel ist die Integration beliebiger Software-Komponenten bei zeitsynchroner Anbindung an beliebige I/O-Devices im Online- als auch im Offline-Betrieb durch transparente Zugriffe auf bestehende Datenaufzeichnungen. Reale Hardware wird dadurch nicht benötigt.

Quelle: [http://www.audi-electronics-venture.de/aev/brand/de/leistungen/Entwicklungstools/adtf.html abgerufen am 28.09.12](http://www.audi-electronics-venture.de/aev/brand/de/leistungen/Entwicklungstools/adtf.html%20abgerufen%20am%2028.09.12)

### Driver Assistance Application and Safety System Development with EB Assist ADTF

Software plays an important role in today’s vehicle development. Driver assistance and vehicle safety systems are state-of-the-art and a must for a competitive car manufacturer. The EB Assist Automotive Data and Time-Triggered Framework (ADTF) supports the software developer in creating new functionalities with an extensive software development kit for driver assistance solution.

## The Automotive Data and Time-Triggered Framework

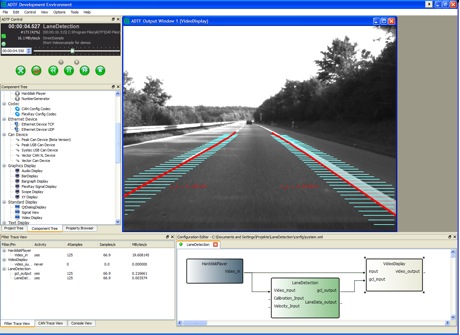
EB Assist ADTF, the Automotive Data and Time-Triggered Framework, is a flexible tool for the development of new functions in the car. The modular system provides together with standard components a solid basis with open interfaces. With the platform independent software development kit new functions can be efficiently implemented. To protect your intellectual property the framework additionally offers the possibility to exchange software components in binary form.

EB Assist ADTF is able to capture asynchronous data from different sensor sources and provides standard components for data recording and interpretation of LIN, MOST, CAN and FlexRay bus systems. Besides data recording, the framework offers tools for real-time data playback, data handling, processing and visualization in the lab as well as online in the car. To support data exchange with proprietary tools, a so called Streaming Library is available.

EB Assist ADTF simplifies the development process, especially the cooperation between OEMs and suppliers. The initial development of the framework has been driven by a major German OEM. Currently the product is in use by several renowned OEMs and Tier 1 suppliers.

## EB Assist ADTF is:

* Easy exchange of data and components
* Flexible and extendable set of modules
* Live visualization of data and results
* Comfortable GUI for configuration and control
* Real-time data recording, streaming and playback



#### How EB Assist ADTF works

The infrastructure of EB Assist ADTF provides the basis for the software development cycle for driver assistance functions and supports the engineer in the software testing and verification process. The framework connects a development environment with an interactive work environment. Without writing a single line of code developers are able to create new configurations by using the graphical user interface and existing modules. The signal data flow between software components is defined by drag and drop and can be executed immediately, so that the effects are instantly visible. The provided examples, libraries and tool boxes facilitate the development of your own driver assistance and active safety software modules which can easily be integrated into the framework.

EB Assist ADTF describes a binary standard. Functional interfaces and data formats are open for developers. EB Assist ADTF is available for Microsoft Windows and Linux operating systems.

Quelle: [http://ww.automotive.elektrobit.com/home/driver-assistance-software/eb-assist-adtf.html abgerufen am 28.09.2012](http://ww.automotive.elektrobit.com/home/driver-assistance-software/eb-assist-adtf.html%20abgerufen%20am%2028.09.2012)

# Fahrerssistenzsysteme einfacher entwickeln und validieren

06.02.12 | Autor / Redakteur: Emil Näpflein \* / Thomas Kuther

[PDF](http://www.elektronikpraxis.vogel.de/index.cfm?pid=5404) | [Weiterempfehlen](JavaScript:openPopUp('/index.cfm?pid=5417&pk=352100&ct=10')) | [Merken](http://www.elektronikpraxis.vogel.de/index.cfm?pid=5404) | [Drucken](http://www.elektronikpraxis.vogel.de/index.cfm?pid=5416&pk=352100&print=true&printtype=article)

[](http://www.elektronikpraxis.vogel.de/index.cfm?pid=11180&pk=440647&type=article&fk=352100)

Fahrerassistenzsysteme: Die extrem hohen Anforderungen in Sachen Entwicklung und Validierung von Systemen wie dem Kreuzungsassistenenten von Audi lässt sich mit den richtigen Tools vereinfachen (Bild: Audi)

Immer leistungsfähigere Fahrerassistenzsysteme erfordern einen immer höheren Aufwand in Sachen Entwicklung und Validierung. Mit den richtigen Tools lässt sich dieser Aufwand aber in Grenzen halten.

Die Liste der Fahrzeuge mit Fahrerassistenzsystemen wird immer länger. Diese Systeme sollen den Fahrer in bestimmten Situationen unterstützen und so insbesondere die Sicherheit und den Fahrkomfort erhöhen. Um diese Assistenzfunktion erfüllen zu können, sind verschiedene Sensoren zum Wahrnehmen der Umwelt erforderlich. Die Informationen dieser Sensoren werden aufbereitet und mit komplexen Algorithmen ausgewertet, sodass das Assistenzsystem dem Fahrer im entscheidenden Moment Hinweise geben oder ihn bei der Bewältigung einer Situation – gegebenenfalls auch durch einen Eingriff in die Fahrzeugführung – unterstützen kann.

### Die ersten Assistenzsysteme waren Spezialentwicklungen

Die ersten Assistenzsysteme sind als Spezialentwicklungen der jeweiligen Fahrzeughersteller und Zulieferer entstanden. Sie haben mit eigenen Tools eine passende Entwicklungsumgebung geschaffen. Die schnelle technische Entwicklung, die steigende Komplexität der Anwendungen und die zunehmende Verlagerung von Funktionalität zu den Zulieferern, haben den Aufwand für die Pflege und Erweiterung der Entwicklungsumgebungen bei den Automobilherstellern und Zulieferern indes rapide ansteigen lassen.

### Synergieeffekte dank freigegebener Entwicklungsumgebung

Aus diesen Gründen war es eine konsequente Maßnahme von Audi Electronic Venture (AEV), ihre Entwicklungsumgebung „Automotive Data and Time-triggered Framework (ADTF)“ auch für andere Automobilhersteller und Zulieferer zur Verwendung freizugeben. Seit 2008 wird ADTF unter der Bezeichung „EB Assist“ von Elektrobit sowohl für Windows als auch für Linux lizenziert. Mittlerweile gibt bereits mehr als 500 Installationen, mit denen mittlerweile eine Vielzahl von Assistenzsystemen erfolgreich entwickelt worden sind. Dank der breiten Verwendung durch alle, an der Entwicklung von Fahrerassistenzsystemen Beteiligten, ergeben sich Synergieeffekte, die sich letztlich auf die Gesamtkosten positiv auswirken. Alle Beteiligten können Komponenten in Quellform – wenn Knowhow geschützt werden soll auch in Binärform – austauschen und problemlos in ihrer eigenen Umgebung einsetzen. So lassen sich schon frühzeitig SIL-Tests mit zugelieferten Komponenten durchführen. Kostspielige Tests auf der Straße können bei gleicher Qualität reduziert werden. Das Installationspaket von ADTF enthält die folgenden Komponenten:

* C++-Entwicklungsumgebung (SDK),
* ADTF Runtime System mit Konfigurationseditor sowie
* Dokumentation und Beispiele.

### ADTF-Softwarekomponenten für eigene Anwendungen

Das SDK für Microsoft Visual Studio C++ und Linux GCC ermöglicht es, verschiedene ADTF-Softwarekomponenten für eigene Anwendungen zu entwickeln. Die Funktionalität stellen C++-Klassen bereit. Und durch die umfangreiche Klassenbibliothek lassen sich Komponenten und Anwendungen weitgehend unabhängig vom genutzten Betriebssystem entwickeln. Für die GUI-Entwicklung wird QT verwendet, für grafische Darstellungen OpenGL. Vorhandene Debug-Versionen der Bibliotheken unterstützen die Entwicklung, so liefert z.B. der Debugger von Visual Studio zusätzliche Informationen.

### Tools zum Import, Export und Wandeln von Daten

|  |
| --- |
| [Bild 1: Das ADTF Runtime System stellt die Ablaufumgebung für die entwickelten Komponenten bereit](http://www.elektronikpraxis.vogel.de/index.cfm?pid=11180&pk=440648&type=article) |
| Bild 1: Das ADTF Runtime System stellt die Ablaufumgebung für die entwickelten Komponenten bereit |

Das ADTF Runtime System (Bild 1) stellt die Ablaufumgebung für die entwickelten Komponenten bereit. Wird es gestartet, so lädt es alle notwendigen Komponenten entsprechend dem Inhalt verschiedener XML-Konfigurationsdateien. Der Start erfolgt stufenweise, damit sich Kern, Systemservices und Applikationskomponenten in der richtigen Reihenfolge initialisieren. Über die Lizenz wird gesteuert, ob nur Basisfunktionalität zum Ausführen von ADTF Anwendungen oder aber die volle Funktionalität als Entwicklungsumgebung zur Verfügung steht. Die Entwicklungsumgebung enthält auch eine ganze Reihe von Tools mit denen Daten importiert, exportiert und umgewandelt werden können.

Mit dem Konfigurationseditor kann man aus vorhandenen Modulen über die grafische Oberfläche einfach Filtergraphen erstellen, die eine ADTF-Anwendung repräsentieren. Diese Filtergraphen werden in Konfigurationen abgelegt die man per Mausklick umschalten kann. Die mit vielen Programmbeispielen versehene Dokumentation hilft beim Einstieg in das Thema ADTF. Die Beispiele lassen sich häufig auch als „Templates“ für die eigenen Komponenten verwenden und können dann direkt in der Entwicklungsumgebung übersetzt und mit dem Runtime System ausprobiert werden.

### Einheitliches Datenformat für alle relevanten Datenströme

Ein zentraler Bestandteil von ADTF ist ein standardisiertes Containerformat für alle relevanten Datenströme. Dadurch wird der Datenaustausch zwischen den am Entwicklungsprozess Beteiligten erheblich erleichtert. Aufzuzeichnende Datenströme, z. B. Video und CAN, werden vom mitgelieferten Harddisk\_Recorder zeitsynchron aufgezeichnet. Diese lassen sich dann auch wieder zeitsynchron mit dem mitgelieferten Harddisk\_Player abspielen, um neue Algorithmen zu testen oder mit SIL- und HIL-Tests die Qualität von Komponenten zu validieren. Neben Standarddatenformaten für die Datenströme kann man aber auch ganz neue Formate über eine Data Definition Language (DDL) in XML definieren.

### Beliebig miteinander kombinierbare ADTF-Filter

Eine ADTF-Anwendung wird mit ADTF-Filtern realisiert. Jeder Filter hat mindestens einen Eingabe- oder einen Ausgabe-Pin. Möchte man etwa die Daten einer Videokamera aufzeichnen, benutzt man einen Kamerafilter (Bild 2). Dieser liest die Daten von der Schnittstelle an die die Kamera angeschlossen ist, konvertiert die Bildinformation in ein ADTF-Bildformat und gibt dann zeitsynchron jedes Bild über den Ausgabe-Pin des Kamerafilters als so genanntes MediaSample aus.

|  |
| --- |
| [Bild 2: Eine ADTF-Anwendung wird mit solchen ADTF-Filtern realisiert, wobei jeder Filter mindestens einen Eingabe- oder einen Ausgabe-Pin hat](http://www.elektronikpraxis.vogel.de/index.cfm?pid=11180&pk=440649&type=article) |
| Bild 2: Eine ADTF-Anwendung wird mit solchen ADTF-Filtern realisiert, wobei jeder Filter mindestens einen Eingabe- oder einen Ausgabe-Pin hat |

Das MediaSample ist hier ein Container der beliebige Daten enthalten kann, in diesem Fall ein einzelnes Videobild. Einer oder mehrere verbundene Verarbeitungsfilter nehmen dieses MediaSample an ihren Eingabe-Pins über ein Ereignis (Event) entgegen, verarbeiten die Daten und geben ihre Verarbeitungsergebnisse wiederum weiter, bis am Ende vielleicht ein Ausgabefilter das gesamte Verarbeitungsergebnis in Echtzeit ausgibt. Es können auch 1:N, N:1 und N:N Verbindungen hergestellt werden. Den Kombinationsmöglichkeiten sind dabei fast keine Grenzen gesetzt, solange die Datenformate der verbundenen Pins übereinstimmen.

ADTF Services stellen den Filtern zusätzlich zentrale Funktionalität bereit. So kann man eine gemeinsame grafische Oberfläche für das Parametrisieren und das Bedienen realisieren. Im Prinzip lassen sich Services über ihre bereitgestellten Schnittstellen wie Bibliotheken nutzen. Ein Beispiel für die Nutzung von Filter und Services zeigt Bild 3.

|  |
| --- |
| [Bild 3: Die Nutzung von Filtern und Services am Beispiel des Philosys Labeleditors](http://www.elektronikpraxis.vogel.de/index.cfm?pid=11180&pk=440650&type=article) |
| Bild 3: Die Nutzung von Filtern und Services am Beispiel des Philosys Labeleditors |

### Label Editor als Beispiel für die Anwendung von ADTF

Für die Validierung der Algorithmen von bildbasierten Fahrerassistenzsystemen nutzt man den Vergleich zu den Ergebnissen die der Mensch liefert. Dazu werden die gleichen Videoszenen, die das Assistenzsystem verarbeitet, auch durch den Menschen bewertet und so genannte Ground-Truth-Daten erzeugt („Labeln“). Diese beschreiben dann z.B. die Position von Objekten im Videobild und verschiedene Attribute, wie Informationen über das Objekt selbst und Umweltbedingungen wie Wetterphänomene. Um diese Ground-Truth-Daten zu erfassen benötigt man effiziente Tools, wie den Philosys Labeleditor (Bild 4). Bei ihm wurden die einzelnen Fenster der Oberfläche mit ADTF Services und Qt realisiert. Für die Darstellung des aktuellen Bildes und der Objektmarkierungen wird OpenGL verwendet.

|  |
| --- |
| [Bild 4: Die Bedienoberfläche des auf ADTF basierenden Programms zum Labeln von Videodatenströmen Philosys Labeleditor](http://www.elektronikpraxis.vogel.de/index.cfm?pid=11180&pk=440651&type=article) |
| Bild 4: Die Bedienoberfläche des auf ADTF basierenden Programms zum Labeln von Videodatenströmen Philosys Labeleditor |

### Vorhandene ADTF-Filter lassen sich direkt nutzen

Die Services tauschen Daten mit ADTF Services aus. Die Steuerung der Videowiedergabe erfolgt über die ISampleStream-Schnittstelle des Harddisk\_Players. Die Implementierung mit ADTF hat den Vorteil, dass die direkte Nutzung von bereits vorhandenen ADTF-Filtern bei der Aufbereitung der Videodaten möglich ist. So ist es nicht notwendig, die Daten für die Erzeugung der Ground-Truth Daten nochmal zu konvertieren. Dies spart in Anbetracht der großen Datenmengen nicht nur viel Zeit sondern auch Speicherplatz. Auch können mit entsprechenden Filtern nützliche Zusatzinformationen, etwa aus CAN-Daten, mit in das Videobild eingeblendet werden. Der Ablauf beim „Labeln“ wird durch die automatische Übernahme der Daten aus dem aktuellen Bild mit Extrapolation in das nächste Bild deutlich beschleunigt. Das Ergebnis ist dann eine XML-Datei die für jedes Bild im Video die relevanten Objekte mit den jeweiligen Attributen beschreibt. In der Validation werden dann diese Daten in SIL- und HIL-Tests automatisch verglichen.

### Erfahrungen aus vielen Kundenprojekten im ADTF-Umfeld

In die Entwicklung des Labeleditors sind die umfangreichen Erfahrungen von Philosys aus einer Vielzahl von Kundenprojekten im ADTF-Umfeld eingeflossen. So wurden für ihn verschiedene Filter und Services, mit oder ohne GUI-Oberfläche (Qt und OpenGL) entwickelt. Das gesammelte Knowhow ist auch die Basis für das breite Dienstleistungsangebot, wie ADTF-Softwareentwicklung, ADTF-Hotline und kundenspezifische ADTF-Workshops. Philosys unterstützt so Automobilhersteller und deren Zulieferer nicht nur beim Einstieg in das Thema ADTF, sondern sorgt durch Übernahme von ADTF-Entwicklungs- und -Validierungsaufgaben mit ADTF-Tools auch dafür, dass sich die Assistenzsystementwickler auf ihre Kernkompetenz – die Entwicklung und Verbesserung neuer Algorithmen – konzentrieren können.

### Entwicklung auf mehrere Unternehmen verteilbar

Mit ADTF steht den Herstellern von Assistenzsystemen ein leistungsfähiges Tool zur Verfügung, das den Bereich von der Forschung bis zum fertigen Fahrerassistenzsystem abdecken kann. Aufgrund der festgelegten Schnittstellen lassen sich Teile des gesamten Entwicklungsprozesses auf mehrere Unternehmen verteilen. Durch den möglichen Austausch von Komponenten und Daten können Mehrfachentwicklungen für Tools vermieden und so Kosten gespart werden und die Qualität unternehmensübergreifend einheitlich geprüft werden. Dieses einheitliche Konzept erleichtert es letztlich auch völlig neue Ideen für Assistenzsysteme zu verwirklichen.

Quelle: <http://www.elektronikpraxis.vogel.de/embedded-computing/articles/352100/index2.html> abgerufen am 28.09.2012