



**BOSCH**

# Softwaretest von verteilten Echtzeitsystemen im Automobil anhand von Kundenspezifikationen

S. Jovalekic <sup>1)</sup>, G. Martinek <sup>1)</sup>, Th. Okrusch <sup>2)</sup>

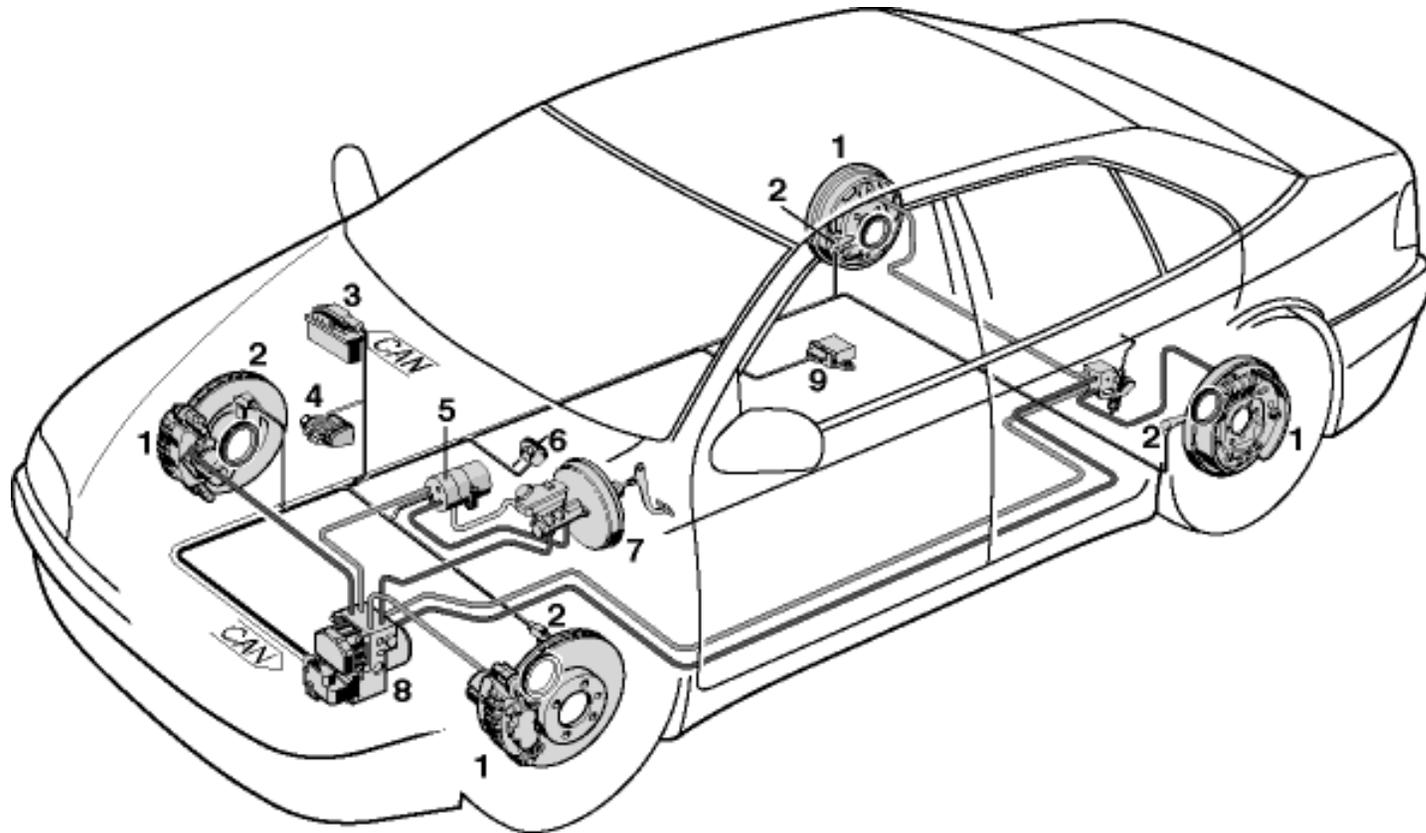
<sup>1)</sup> FH Albstadt-Sigmaringen, 73458 Albstadt

<sup>2)</sup> Robert Bosch GmbH, Abstatt

# Gliederung

- Einleitung
- Datenverkehr und Informationsfluss auf dem CAN Bus
- Softwarearchitektur der Testautomatisierung
- Anforderungen und Architektur eines Werkzeuges
- Ergebnisse und Ausblick

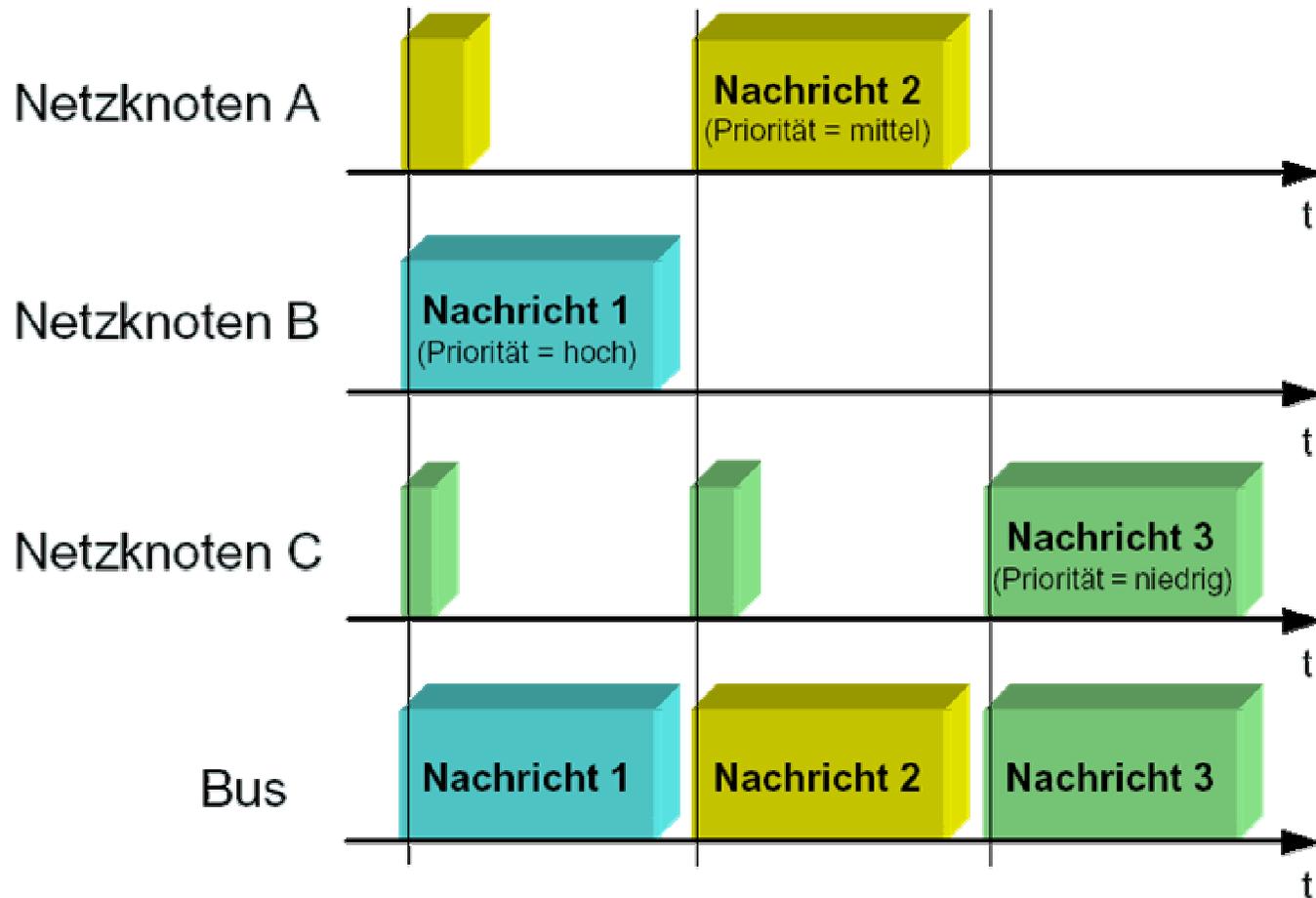
# Einleitung



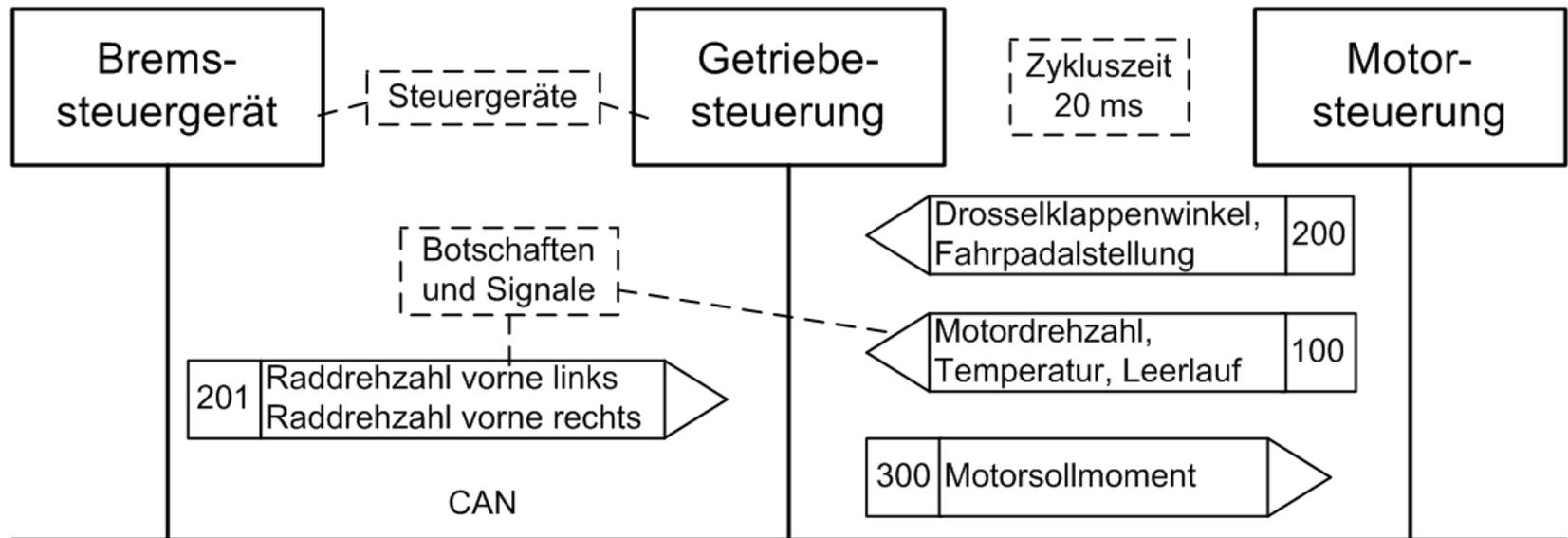
# CAN Eigenschaften

- Vernetzung von Steuergeräten mit geringer Entfernung.
- Serielle Datenübertragung über eine abgeschirmte Zweidraht-Leitung.
- Die Botschaften sind maximal 8 Datenbytes lang.
- Die Baudrate beträgt 1 Mbit/s bei 30m und 125 kbit/s bei 500m Buslänge.
- Inhalt einer Botschaft wird durch einen busweit eindeutigen Identifier gekennzeichnet.
- Der Identifier legt auch die Priorität der Botschaft fest.
- Die Zuteilung des Busses erfolgt mit bitweiser Arbitrierung.
- Die Steuergeräte kommunizieren nach dem Multi-Master Prinzip.
- leistungsfähige Fehlererkennung und Fehlerbehandlung.

# Datenverkehr auf dem CAN Bus



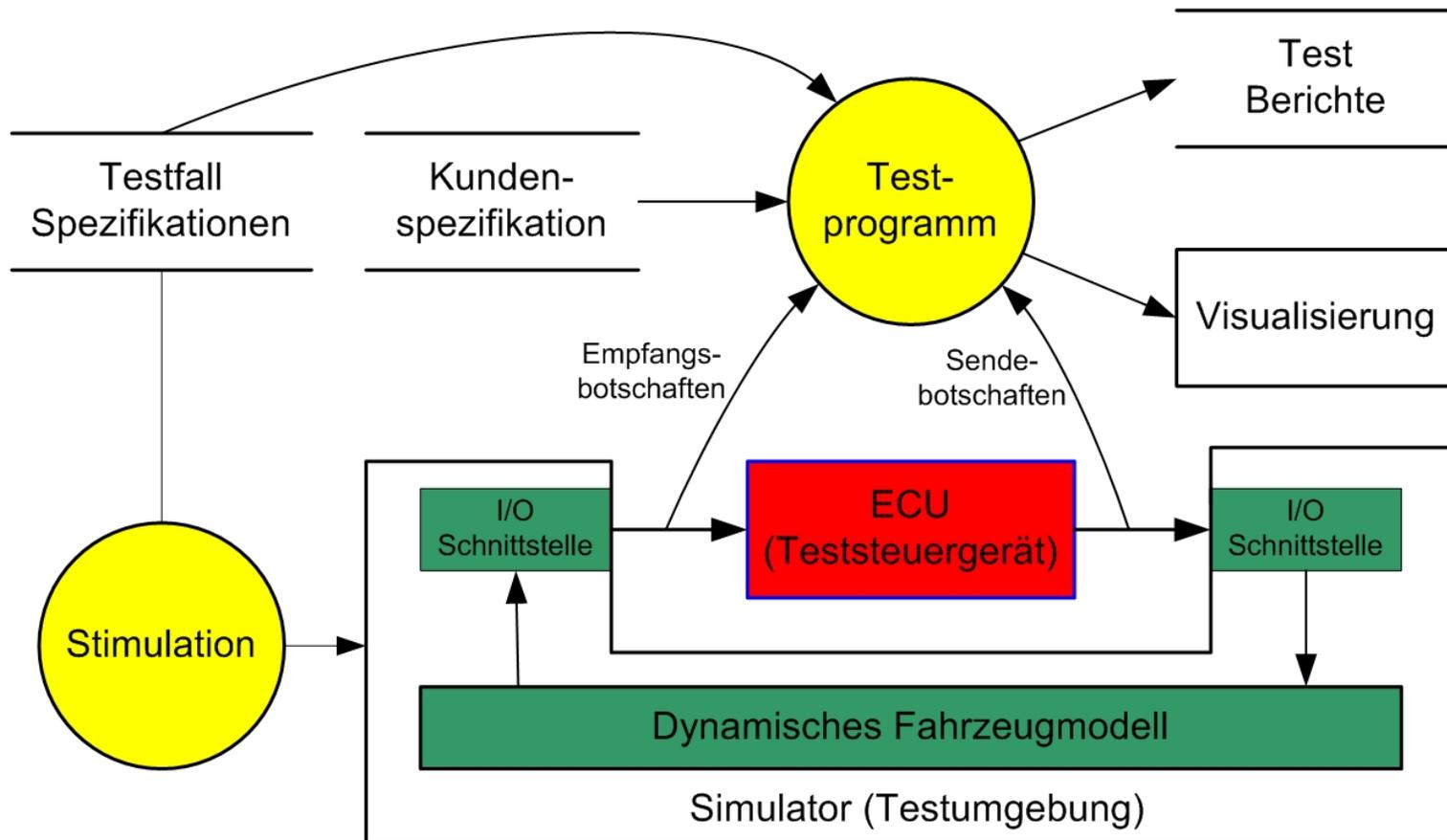
# Informationsaustausch über den CAN Bus



# Automatisiertes Testen

- Motivation: Manuelles Testen ist mühsam und fehlerbehaftet
- Vorteile der Automatisierung:
  - Steigerung der Testproduktivität/Testüberdeckung
  - Tests sind jederzeit und ohne hohen Aufwand durchführbar
  - Tests sind exakt wiederholbar (gleiche Eingaben)
  - Ermöglicht komplexe Tests ( > 1000 Testfälle)
- Aspekte der Automatisierung:
  - Erzeugung von Testeingaben und erwarteter Ergebnisse
  - Testdurchführung ohne manuellen Eingriff
  - Auswertung (erfolgreich / nicht erfolgreich)

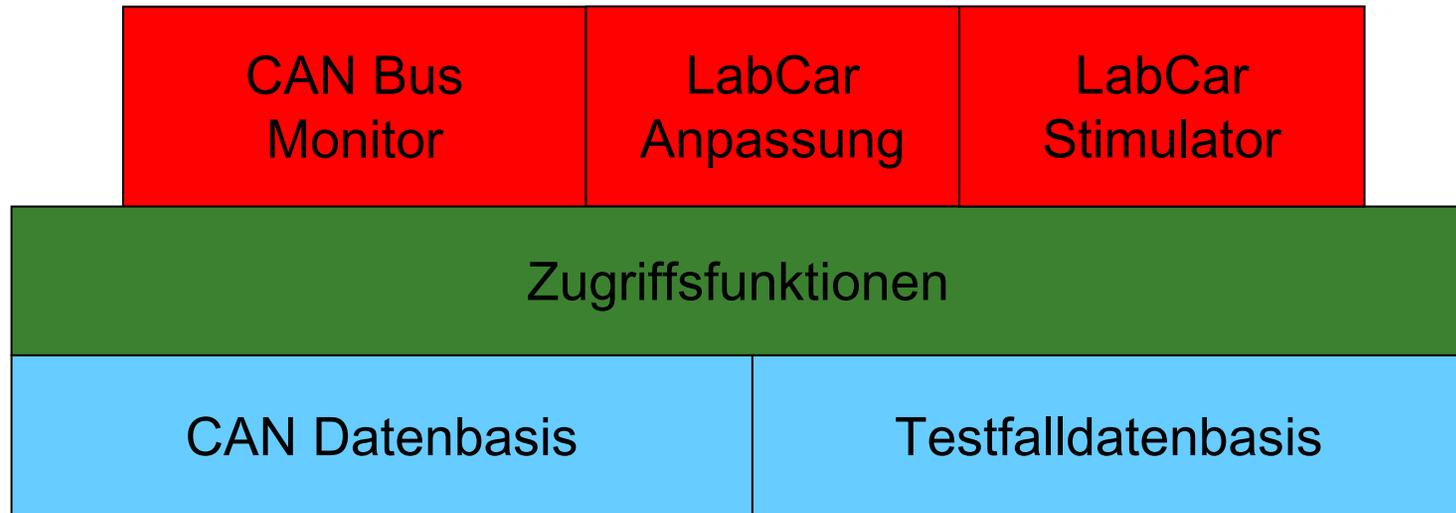
# Softwarearchitektur der Testautomatisierung



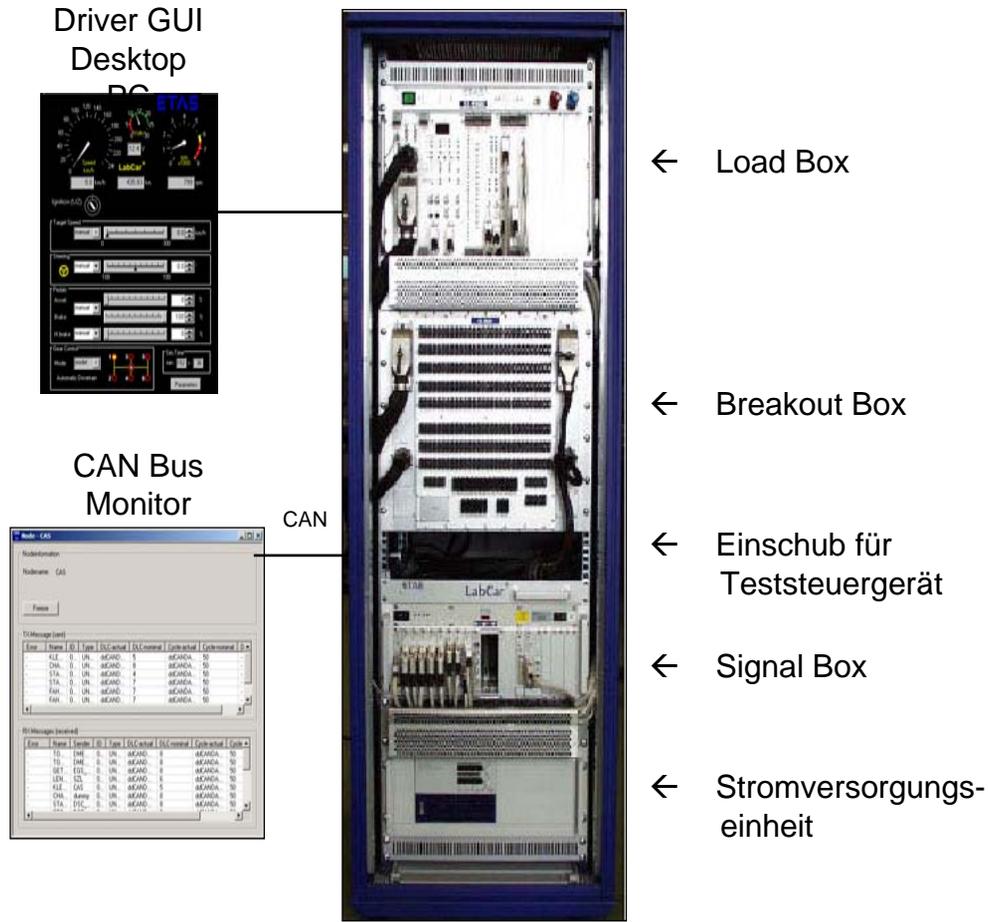
# LaborAuto als Testumgebung



# Grobarchitektur der Testwerkzeugkette



# Einsatzumgebung für das Testwerkzeug



# Anforderungen an den CAN Bus Monitor

- Automatischer Vergleich von Ist- und Sollwerten der Signale entsprechend der CAN Datenbasis
- geeigneter Darstellung erkannter Fehler
- Verwaltung von Testprojekten
- Übersicht der Busteilnehmer um den Testknoten im Netzwerk.
- Definition von Botschaftsfiltern zur Datenreduktion.
- Offline Analysen bereits erfasster Daten.
- Langzeit Analysen über einen Zeitraum bis 20 Minuten.
- Report Generierung am Ende einer Analyse.



# Zusammenfassung und Ausblick

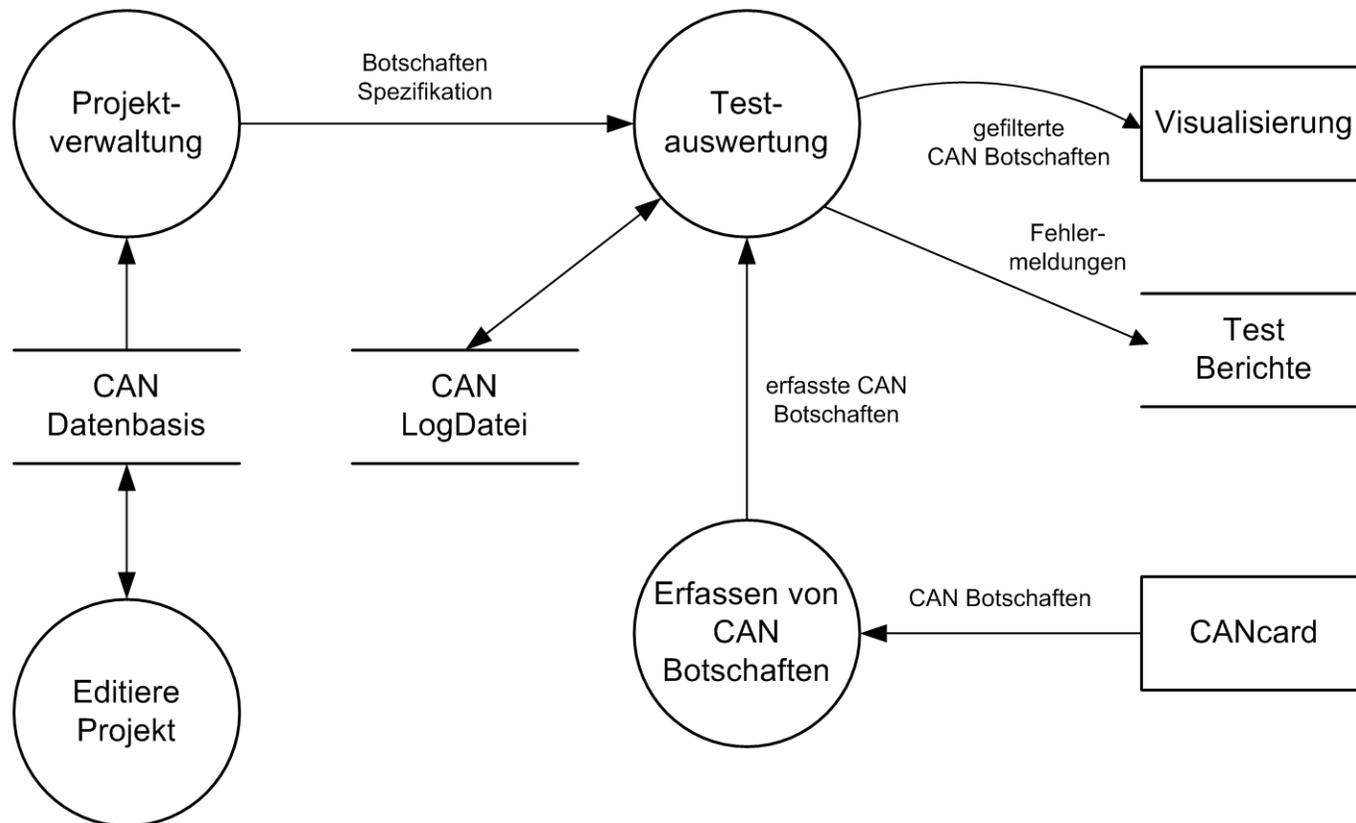
- Zeitersparnis bei den Tests wird erwartet
- Vollständige Überprüfung der tatsächlich gesendeten Daten mit denen aus der Spezifikation
- Vorteile durch Gliederung der Software in Schichten mit definierten Schnittstellen:
  - Änderbarkeit
  - Optimierung
  - Wiederverwendbarkeit von gesamten Subsystemen
  - Vereinfachte Verteilung und Zuordnung der Aufgaben
- Externe Steuerung des Monitors

---

Danke für Ihre Aufmerksamkeit !

Fragen ?

# Analyse des CAN Bus Monitors



# Ansichten des CAN Bus Monitors

The screenshot displays three main components of the CAN Bus Monitor software:

- Treeview:** A hierarchical tree structure on the left showing various CAN nodes. The selected node is `NAV_GPS1 (0x34a)`, which is expanded to show sub-nodes like `GAP_AVL_TARO` and `CHKSM_DCRN_BRP_ACC`.
- Traceview:** A central window displaying a table of CAN messages. The table has columns for Error, Time, Chn, ID, Name, Dir, DLC, Data, Type, and TXNode. It lists various messages such as `Airbag_1`, `Bremse_1`, `Motor_1`, `ACC_1`, `Gateway_Komfort_1`, `Getriebe_1`, `Motor_5`, `Bremse_3`, `Bremse_5`, `Kombi_3`, `Motor_Flexia_neu`, `Bremse_2`, and `EPB_1`.
- RX-Messages (received):** A table at the bottom showing received messages. It has columns for Error, Name, Sender, ID, Type, DLC-actual, DLC-nominal, Cycle-actual, and Cycle. The messages listed include `DME...`, `EGS...`, `SZL...`, `CAS...`, `dummy...`, and `DSC...`.