

>>> DIAGNOSE ALS TEIL EINES E/E-ENTWICKLUNGSPROZESSES

Diagnosedaten effizient erfassen, spezifizieren und in Software umsetzen

In gängigen Entwicklungsprozessen werden Diagnosebeschreibungen und deren Software-Umsetzung nicht nur durch unterschiedliche Personen und Rollen in Unternehmen bearbeitet, sondern auch durch verschiedene Autorenwerkzeuge unterstützt. Die Konsistenz zwischen beidem muss manuell gewährleistet werden und ist damit aufwendig und fehleranfällig. Durch einen modellgetriebenen Entwicklungsansatz wird es möglich, die Diagnose gemeinsam mit dem gesamten E/E-System eines Fahrzeugs zu beschreiben.

In Projekten rund um die Fahrzeugdiagnose lässt sich regelmäßig beobachten, dass die Diagnosespezifikation und die Software-Entwicklung weitgehend entkoppelt

sind. Die Zuständigkeiten für die Diagnosespezifikation und Applikationssoftware-Entwicklung sind heute oft auf verschiedene Personen, Rollen und Abteilungen verteilt. Die Mitarbeiter erfüllen

die notwendigen Aufgaben mit unterschiedlichen Werkzeugen und verwalten die Arbeitsergebnisse isoliert voneinander, es besteht keine gemeinsame Datenbasis. Notwendige Abstimmungen



erfolgen oft nur ad hoc und ohne einen standardisierten Datenaustausch. Der Informationsaustausch zwischen den verschiedenen Bereichen erfolgt damit informell über E-Mails oder den Austausch von Dokumenten. Die eigentlichen Daten müssen manuell in die verschiedenen Werkzeuge eingepflegt werden.

Diese Beobachtungen überraschen wenig, wenn man bedenkt, wie sehr die Aufgabenfelder der Diagnose- und der Software-Entwicklung in der Praxis auseinanderfallen. Ein erfolgreiches Diagnosedesign muss den gesamten Systemlebenszyklus berücksichtigen. Außerhalb der eigentlichen Systementwicklung muss die Diagnoselösung auch die Anwendungsfälle der Fahrzeugproduktion und des Kundendienstes unterstützen. Experten aus diesen Bereichen betrachten die Fahrzeugdiagnose aus unterschiedlichen Blickwinkeln. Für Produktions- und Kundendienstexperten liegt ein wesentlicher Fokus auf Protokolldetails für die Parametrierung der unterschiedlichen Tester in Produktion und Kundendienst. Für Systementwickler steht hingegen die innere Ausgestaltung und Funktionsweise der Fahrzeug-Software im Mittelpunkt des Interesses. Beides ist für die Arbeit der jeweils anderen Seite weitestgehend irrelevant.

Abhängigkeiten zwischen der Diagnose und Software bestehen aber! Es können Inkonsistenzen entstehen, wenn beispielsweise die Datentypdetails der Diagnosespezifikation nicht zur umgesetzten Applikationssoftware passen. Diese werden bestenfalls bereits bei der Konfiguration der in AUTOSAR definierten Basissoftware-Komponenten wie Diagnostic Event Manager (DEM) und Diagnostic Communication Manager (DCM) erkannt, im schlechtesten Fall aber erst bei der Inbetriebnahme des Steuergeräts oder sogar erst im Feld. Hier greift die alte Regel: Je später ein Fehler erkannt wird, desto teurer und aufwendiger ist seine Behebung. Wenn es gelingt, die Beschreibung der Applikationssoftware zusammen mit den Diagnoseinhalten in einem Werkzeug zu erstellen, dann können die genannten Inkonsistenzen und die damit verbundenen Folgeprobleme vermieden werden. Eine effiziente Methode zum Meistern der beschriebenen Herausforderungen bietet die enge Verzahnung

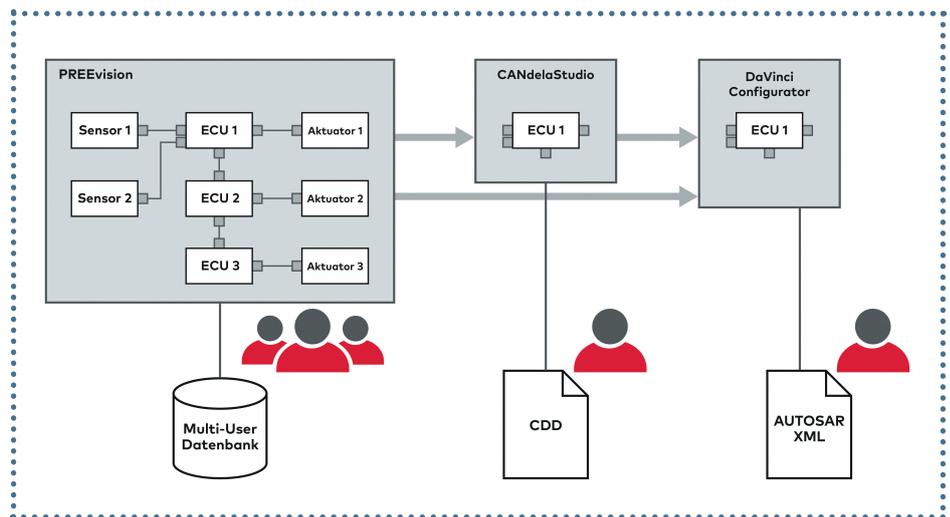


Bild 1: Die Multiuser-Umgebung von PREEvision erlaubt einen systemweiten Blick, wohingegen die nachgelagerte Toolkette jeweils den Fokus auf ein einzelnes Steuergerät legt. (© Vector Informatik)

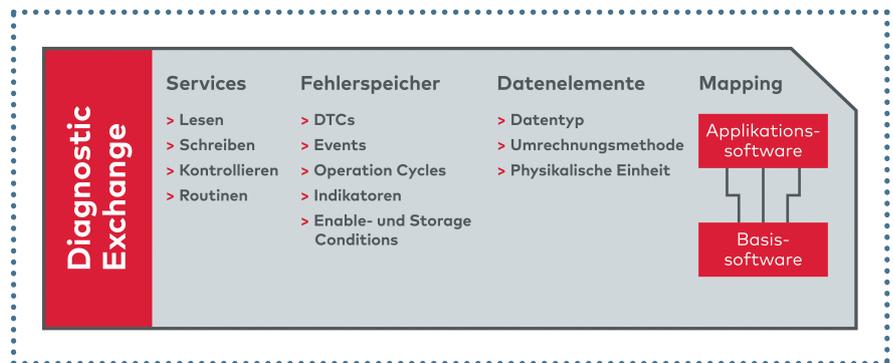


Bild 2: Elemente des Austauschformats. (© Vector Informatik)

zweier Werkzeuge aus dem Produktportfolio von Vector: das modell- und datenbankbasierte Werkzeug PREEvision lässt sich für die Diagnose- und Software-Entwicklung einsetzen und tauscht alle relevanten Daten mit CANdelaStudio aus, dem Vector-Werkzeug zum Erstellen und Bearbeiten formaler Diagnosespezifikationen.

Vorteile modellbasierter Werkzeuge

In der Elektrik-/Elektronikentwicklung (E/E) vieler Automobilhersteller werden die Vorteile des modellbasierten Werkzeugs PREEvision immer intensiver genutzt. Ein übergreifendes E/E-Datenmodell ermöglicht Herstellern, die fachliche Durchgängigkeit zwischen Systemanforderungen, Architektur- und Designebene für Software, Netzwerk und Komponenten der Hardware sowie Leitungssatz und Geometrie zu gewährleisten.

Sie können das gesamte Beziehungswissen zwischen den fachlichen Ebenen und Elementen in diesem gemeinsamen Datenmodell beschreiben (Bild 1). Betrifft eine fachliche Änderung mehrere Ebenen, so wird die Änderung für alle betroffenen Entwickler unmittelbar ersichtlich und der einzelne Anwender wird automatisiert darauf hingewiesen. Dadurch werden mögliche Konflikte frühzeitig erkannt und ausgeräumt. Der Anwender kann zeitliche Abfolgen und Zusammenhänge durch die Nutzung von Versionierungsmechanismen zuverlässig nachverfolgen. Darüber hinaus steht eine umfangreiche Unterstützung für das Management von Varianten zur Verfügung. Das ermöglicht dem Anwender, die Gleichteile aber auch Unterschiede von verschiedenen Varianten in einer gemeinsamen Datenbasis zu beschreiben. In der Modellsicht oder für den Datenexport können dann einzelne Varianten aktiviert werden. »

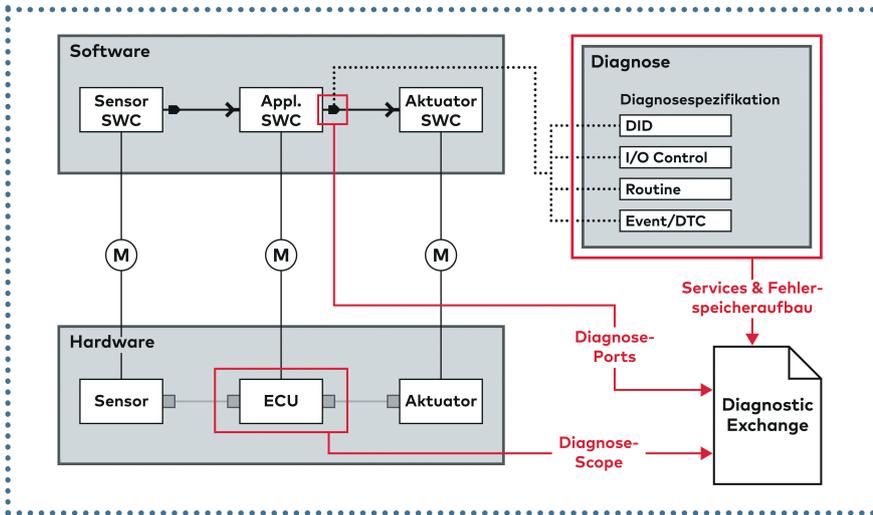


Bild 3: Die Zusammenhänge zwischen Software-, Hardware- und Diagnose-Design und wie daraus die Informationen für den Diagnoseautausch abgeleitet werden (M = Mapping). (© Vector Informatik)

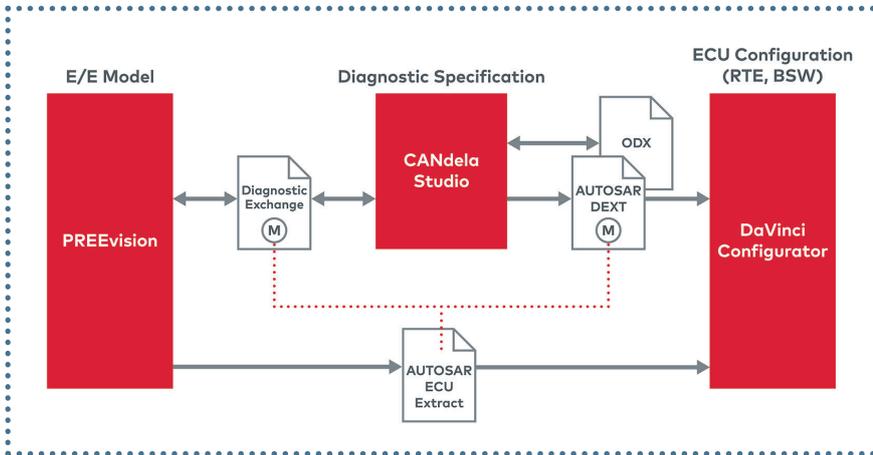


Bild 4: Übersicht der beschriebenen Vector Toolkette für Diagnose (M = Mapping). (© Vector Informatik)

Abhängigkeiten zwischen Diagnose- und Softwaremodellierung

Welche Abhängigkeiten ergeben sich überhaupt zwischen der Steuergeräte-Software und der Diagnosebeschreibung? In vielen Autorenwerkzeugen für die Diagnose werden Diagnosefunktionen eines Steuergeräts aus dem Blickwinkel eines Diagnostesters beschrieben. Der Fokus einer solchen Beschreibung liegt dabei auf den Diagnosefunktionen, die ein externes Testwerkzeug über die Diagnoseschnittstelle des Fahrzeugs nutzen kann. Dazu gehören zum Beispiel Informationen darüber, welche Diagnostic Trouble Codes (DTC) ein Tester aus dem Fehlerpeicher eines Steuergeräts auslesen kann oder welche Data Identifiers (DID) ein Steuergerät zum

Lesen und Schreiben von Daten zur Verfügung stellt. Darüber hinaus werden aber auch Detailinformationen beschrieben, die für die Anzeige im Testgerät von großer Bedeutung sind, zum Beispiel Datentypen, Umrechnungen und physikalische Einheiten der einzelnen Datenobjekte von DIDs und Routinen.

In der Applikationssoftware muss es nun eine Realisierung durch einen Diagnoseport, also die konkrete Ausprägung eines Softwareports, geben. Für ein Datenobjekt eines DIDs ist das ein einzelnes Datenelement eines Diagnoseports. Auch dieses Datenelement wird konkretisiert durch einen Datentyp mit Umrechnung und physikalischer Einheit. Allerdings werden in der Software diese Details durch die AUTOSAR-konforme Modellierung auf mehrere Objekte verteilt, wohingegen sie in vielen gängigen

Diagnosebeschreibungformaten durch ein einzelnes Objekt beschrieben werden. Trotz der unterschiedlichen Granularität und Beschreibungsarten müssen die enthaltenen Informationen aber konsistent sein. Wenn die Diagnose gemeinsam mit der Software in einem Werkzeug definiert wird, ist es möglich Software und Diagnose redundanzfrei zu beschreiben. Die Datentypen werden dann wie gehabt in der Software beschrieben. Die Diagnosebeschreibung referenziert diese Datentypen, verwendet also dieselben Objekte wieder. Alternativ ist es auch möglich, zuerst die Diagnoseinhalte zu beschreiben und die gemeinsamen Detailinformationen für die Software-Umsetzung zu nutzen.

Neben der Notwendigkeit, gemeinsame Datenbeschreibungen konsistent zu halten, ergeben sich aus den Diagnosespezifikationen auch Anforderungen an die Applikationssoftware selbst. Definiert die Diagnosespezifikation beispielsweise eine Routine, so muss es in der Applikation eine Softwarekomponente geben, die diese Routine implementiert. Definiert die Diagnosespezifikation einen DTC, so muss die Applikation eine Monitorfunktionalität aufweisen, die das Auftreten des damit verbundenen Fehlerereignisses feststellen kann. AUTOSAR stellt dabei spezifische, strukturelle Anforderungen an die realisierenden Softwarekomponenten. Ein I/O-Control erfordert beispielsweise die Abbildung der kontrollierbaren Daten durch einen Record-Typ. Eine Routine wiederum erfordert ein Client/Server-Interface mit Start-, Stop- und Request-Result-Operations. Auch hier hilft es, die Strukturen nur einmal zu definieren und gleichermaßen für die Software und Diagnose zu nutzen.

Konkrete Umsetzung in einer Werkzeugkette

Um den beschriebenen Problemen zu begegnen, wurde PREEvision um eine eigene Modellierungsebene für die Diagnose erweitert. Diese Diagnoseebene ermöglicht es, Diagnoseobjekte systemweit zu beschreiben und an die Applikationssoftware anzubinden. Der Anwender kann hier DIDs mit ihren Datenobjekten, I/O-Controls und Routinen definieren. Für diese Objekte können



auch einige ergänzende Eigenschaften, wie Identifier oder die verwendeten Diagnosedienste, beschrieben werden. Des Weiteren können auch relevante Teile des Eventhandlings gemäß AUTOSAR modelliert werden. Dazu gehören Objekte wie DTCs, DiagnosticEvents und DiagnosticOperationCycles, um nur einige zu nennen. Für die Konkretisierung der Diagnosebeschreibung sind nach wie vor Autorenwerkzeuge wie CANdelaStudio unerlässlich.

Eine Toolkette aus PREEvision und CANdelaStudio bietet deswegen einen bidirektionalen Datenaustausch (Bild 2) und unterstützt verschiedene Zusammenarbeitsmodelle. Auch ein Datenrückfluss nach PREEvision ist möglich. Dadurch kann der Anwender einfach Bestandsdaten aus der Diagnose übernehmen und als Startpunkt für seine Software-Realisierung nutzen. Für den Datenaustausch wurde ein dateibasierter Ansatz gewählt, was einen Austausch über Organisationsgrenzen hinweg ermöglicht. Die Import- und Exportmöglichkeiten in CANdelaStudio erschließen ein breites Spektrum in der Zusammenarbeit mit Zulieferern und in der Ankopplung an verschiedene Toolketten. Die Austauschdatei enthält sowohl die eigentlichen Diagnoseobjekte, aber auch die zugehörigen Details, die aus der referenzierten Software abgeleitet sind.

Obwohl die Modellierung der Diagnoseobjekte systemweit durchgeführt wird, finden sich in der Austauschdatei nur die Informationen für ein einzelnes Steuergerät. Abhängig von der Software- und Hardwaremodellierung wird ermittelt, welche Diagnoseports auf einem Steuergerät überhaupt zur Verfügung stehen. Diese Information ist die Basis für die Zusammenstellung der Diagnoseobjekte für den Datenaustausch (Bild 3). Es gibt noch einen weiteren Aspekt, der für den nachgelagerten Prozess hilfreich ist: Die Information über die Zuweisung des Diagnoseobjekts zum Diagnoseport in der Applikationssoftware wird im Datenaustausch mittels einer Referenz transportiert. In einem Konfigurationswerkzeug für Basissoftware muss der Integrator nun nicht die Verbindungen zwischen den Ports der Basissoftware und denen der Applikationssoftware ziehen. Die Verbindungen können anhand der Referenzen au-

tomatisch ermittelt werden, womit die Fehleranfälligkeit bei gleichzeitiger Zeiterparnis erheblich sinkt.

Diagnosekonformität in der Software sicherstellen

Im AUTOSAR-Standard ist festgelegt, wie Softwarekomponenten für die Diagnose zu beschreiben sind. Wird beispielsweise ein Diagnoseport erzeugt, muss dieser AUTOSAR-konform modelliert sein. Allerdings ist der AUTOSAR-Standard umfangreich, komplex und nicht immer einfach zu verinnerlichen. Häufiges Nachschlagen in den Regelwerken ist gängige Praxis und das Umsetzen des Standards zeitaufwendig und anfällig für Fehler. Dank Toolunterstützung lassen sich diese Daten effizienter und in deutlich höherer Qualität erstellen: Erzeugt man einen Diagnoseport direkt aus dem eigentlichen Diagnoseobjekt, wird dieser Port toolgestützt korrekt und AUTOSAR-konform erzeugt. Führt der Anwender nachträglich Änderungen am Diagnoseport durch, die zu einer Abweichung vom Standard führen, wird er vom Werkzeug direkt darauf hingewiesen.

Vorteile im gesamten Entwicklungsprozess

In PREEvision lassen sich in einem Modell zusätzlich zu Software, Hardware und Kommunikation auch Diagnoseinformationen für ein E/E-System beschreiben. Der Anwender verwaltet alle Daten zentral, auch die Diagnose wird systemweit betrachtet. Aus der Beschreibung von Software-, Hardware- und Kommunikation ergibt sich die Systembeschreibung nach AUTOSAR (Bild 4). Aus demselben Modell werden die Diagnoseinhalte für CANdelaStudio exportiert und dort in der Diagnosespezifikation weiter verfeinert und ergänzt. Ein manueller Abgleich der Diagnosedaten mit der Software ist nicht mehr nötig: In CANdelaStudio durchgeführte Änderungen werden via Datenaustausch in PREEvision zurückgeschrieben, die Software aktualisiert sich damit automatisch. In CANdelaStudio entsteht der Diagnostic Extract (DEXT) und die ODX-Beschreibung. Zum Konfigurieren der

AUTOSAR-Basissoftware werden schließlich Systembeschreibung und DEXT in Werkzeuge wie Vectors DaVinci Configurator eingelesen. Hier ergeben sich mehrere Vorteile:

- Die Diagnoseobjekte müssen nicht manuell mit den Diagnoseports zusammengeführt werden. Über die gesamte Toolkette übertragene Referenzen ermöglichen eine automatische Zuordnung.
- Die Diagnoseports sind bereits AUTOSAR-konform modelliert und benötigen keine Nacharbeit.
- Die Eigenschaften der Diagnoseports passen zu den Eigenschaften der Diagnoseobjekte, es müssen keine Konflikte bezüglich Datentypen oder Software-Strukturen aufgelöst werden.

Über den gesamten Designprozess ergibt sich damit ein geringerer Abstimmungsaufwand, weniger manueller Aufwand und die Reduktion von Fehlerquellen. Die Vorteile beschränken sich aber natürlich nicht auf die Verknüpfung der Diagnose mit der Applikationssoftware: So ließen sich künftig auch aus anderen Modellebenen in PREEvision automatisiert Diagnosedaten oder -abläufe ableiten, wie beispielsweise Prüfabläufe für einen Kabelbruch direkt aus der Beschreibung des Kabelbaums. Der modellbasierte Ansatz von PREEvision bietet hier erhebliches Potenzial für die Zukunft, um Arbeitsaufwände und Fehlerquellen noch weiter zu verringern.

- (oe)
- » www.vector.com



Claudia Gettkandt ist Produktmanagerin PREEvision bei Vector Informatik.



Armin Cont ist Produktmanager CANdelaStudio bei Vector Informatik.



HANSER

Programmieren mit System

Maschinelles Lernen
GRUNDLAGEN UND ALGORITHMEN IN PYTHON
ISBN 978-3-446-45906-0 | € 38,-

LabVIEW für Einsteiger
Ein Einstieg in die Programmierung
ISBN 978-3-446-45906-9 | € 32,-

PROGRAMMIEREN TRAINIEREN
Mit über 120 Workouts in Java und Python
ISBN 978-3-446-45486-6 | € 30,-

Bestellen Sie jetzt unter www.hanser-fachbuch.de



24

Für Autohersteller eröffnen sich mit UWB (Ultra Wideband) neue Komfortfunktionen und verbesserter Diebstahlschutz.

EDITORIAL

3 **Cybersecurity-Defizite als Menetekel**
von Franz Joachim Rossmann.

NEWS

- 6 **Mit schneller Verifikation zu ASIL D**
- 6 **BMW überträgt OSI-Format an ASAM**
- 9 **Von Drohnen aufgezeichnete Verkehrsdaten**
- 9 **ZF und Danfoss entwickeln Leistungsmodule**
- 10 **Neue Software-Organisation bei VW**
- 10 **Batterieprototypen von Lion Smart verfügbar**
- 11 **Entwicklungspartner für Simulation und Validierung**

TITELSTORY

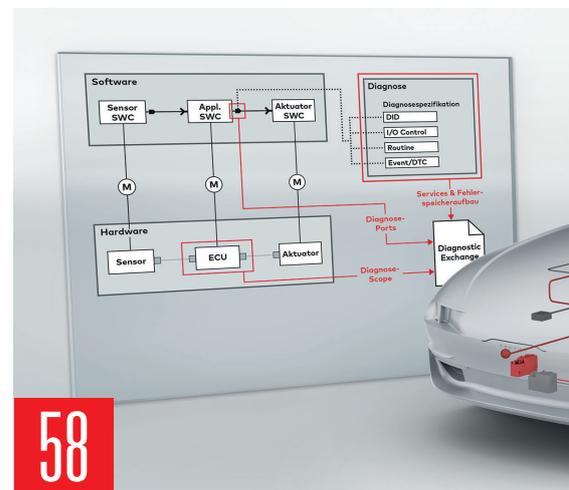
12 **Moderne Sicherheitssoftwareplattform für hochkomplexe Automotive-Anwendungen**
Die MotionWise-Sicherheitssoftwareplattform von TTTech Auto kommt mittlerweile schon bei einer Reihe von Serienentwicklungsprojekten unterschiedlicher Fahrzeughersteller zum Einsatz.

AUTOMATISIERTES FAHREN

- 15 **Prototyping und Data Logging für das Autonome Fahren**
- 16 **Jetzt Aufgaben auslagern, bevor die CPU stirbt!**

CONNECTIVITY

- 20 **Datenhandelsplattform mittels Blockchain-Technologie**
- 24 **Ultra-Wideband-Vernetzungstechnik im Fahrzeug**
- 28 **Kooperative Erlebniswelt**



58

Der modellbasierte Ansatz von PREEvision bietet erhebliches Potenzial Arbeitsaufwände und Fehlerquellen bei der E/E-Entwicklung zu verringern.