Integration der Steuergerätediagnose in den HIL-Test

Dr. Klaus Lamberg, dSPACE GmbH, Paderborn
Dr. Jobst Richert, dSPACE GmbH, Paderborn

Zusammenfassung

Dieser Beitrag beschreibt, wie die Steuergeräte-Diagnose für Hardware-in-the-Loop-(HIL) Anwender verständlich und für den Zugriff aus automatisierten Tests heraus nutzbar in HIL-Systeme integriert werden kann. Vorhandene Implementierungen werden anhand von Beispielen dargestellt. Die dabei in der Praxis auftretenden Probleme werden beschrieben und hinsichtlich ihrer Ursachen analysiert. Abschließend werden Lösungsansätze aufgezeigt, bei denen u.a. auf Basis einer verallgemeinerten HIL-Architektur weiteres Potential für Standardisierung im Umfeld der Steuergeräte-Diagnose und HIL-Simulation besteht.

Summary

This contribution describes, how ECU (Electronic Control Units) diagnostics can be integrated into hardware-in-the-loop (HIL) test systems so that it can be used in automated ECU tests. Existing implementations will be described and the resulting problems will be discussed and analyzed. Finally, an approach will be presented, which – based an a generalized HIL architecture – provides potential for future standardization in the context of ECU diagnostics and HIL simulation.

1 Einführung

Das Thema Steuergerätediagnose gewinnt zunehmend an Bedeutung. Ein Grund dafür ist, dass die Diagnose als bereichs- und unternehmensübergreifendes Thema angesehen werden muss. Ziel der Diagnose heutzutage ist die Unterstützung der gesamten Prozesskette und des gesamten Lebenszyklus' eines Fahrzeugs, von der Entwicklung über die Produktion bis hin zum Service in den Werkstätten. Dies ist nicht nur eine OEM-spezifische Aufgabe, sondern es müssen auch die Steuergerätezulieferer einbezogen werden.

Um die Vielfalt an Diagnosespezifika einzudämmen, einen hohen Wiederverwendungsgrad zu erreichen und letztlich Kosten einzusparen, hat es in den letzten Jahren eine Reihe von Standardisierungsaktivitäten im Diagnosebereich gegeben. Als Beispiel sei hier der Standard ODX (Open Diagnostic Data Exchange Format – ISO 22900) genannt, der als Grundlage für die Erstellung standardisierter Diagnosedatenbasen entwickelt wurde. Diagnosedatenbasen beschreiben beispielsweise, wel-

che Diagnosefunktionalität, sprich „Dienste“, in einem Steuergerät implementiert ist. Gerade ODX ist ein Beispiel dafür, dass durch Standardisierung nicht die technische Komplexität reduziert wird. Im Gegenteil, durch den generischen Anspruch eines Standards wächst im Allgemeinen auch die Komplexität.

Der Anteil der Diagnose am Gesamtumfang der Steuergeräsoftware ist mittlerweile beträchtlich. So entfallen in Motronic-Systemen jeweils ca. 40% der Applikationsparameter, der Funktionen und der Codezeilen auf die Diagnose, und ca. 40-50% der Prozessor-Laufzeit wird für Diagnosefunktionalität benötigt [1]. Dies zeigt in besonderem Maße, dass die technische Komplexität der Diagnose eine zentrale Herausforderung darstellt. Aus diesem Grund ist auf den Test der Diagnosefunktionalität im Rahmen des gesamten Steuergerätetests ein Schwerpunkt der Entwicklungstätigkeiten zu legen.

2 Rolle der Steuergerätediagnose

2.1 Einsatz der Diagnose in Entwicklung, Produktion und Service

Das Thema Diagnose begleitet die Fahrzeugelektronik über ihren gesamten Lebenszyklus hinweg, von der Entwicklung über die Produktion bis hin zum Service. Dabei kommen der Diagnose in den einzelnen Phasen unterschiedliche Aufgaben zu, die im folgenden beschrieben werden.

2.1.1 Diagnoseaufgaben in der Entwicklung

Während der Steuergeräteentwicklung entstehen regelmäßig neue Softwarestände, die in die Steuergeräteprototypen geladen werden müssen. Dieser „Ladevorgang“ erfolgt über das Flashen der betreffenden Speicherbausteine der Steuergeräte. Das Flashen ist dabei eine Teilaufgabe der Diagnose. Die geflashte Software alleine legt die Funktionalität des Steuergeräts jedoch nicht vollständig fest. Dafür muss die Software bzw. die Funktionalität zunächst entsprechend konfiguriert werden. Konfigurationsdaten werden insbesondere benötigt, um länderspezifische Einstellungen (z.B. Warntöne bei Überschreiten der gesetzlich vorgeschriebenen Höchstgeschwindigkeit), fahrzeugspezifische Einstellungen (z.B. Motor- und Getriebeart) und bestimmte Ausstattungsvarianten gemäß der Kundenwünsche vorzunehmen. Dies wird insgesamt auch als Variantenkodierung bezeichnet. Teilweise werden neben Konfigurationsdaten auch Kalibrierdaten wie z.B. Kennlinien, Kennfelder und z.B. Grenzwerte mithilfe von Diagnosetools übertragen. Dies wird aber nur in Einzelfällen mithilfe der Diagnose durchgeführt, da insbesondere für das Kalibrieren (auch: Applizieren) spezielle Applikationswerkzeuge zur Verfügung stehen. Die hierfür verwendeten Schnittstellen und Protokolle unterscheiden sich üblicherweise von denen für die Diagnose.

Eine weitere wesentliche Entwicklungsaufgabe im Zusammenhang mit der Diagnose besteht in der Prüfung der ordnungsgemäßen Funktionalität des Steuergeräts. Dazu gehören insbesondere die Feststellung von Funktionsstörungen, wie z.B. elektrische Kurzschlüsse, Verbindungsunterbrechungen oder Unplausibilität von Signalen sowie die Lokalisierung von Fehlerursachen und die die Störung verursachenden Bauteile. Hierzu dient beispielsweise die Stellglieddiagnose im Rahmen der Hardware-Prüfung. Dabei werden die Ausgänge (Aktoren) des Steuergeräts zu Testzwecken aktiviert. Somit kann im verbauten Zustand überprüft werden, ob die Aktoren ein-

wandfrei funktionieren. Auf den Test der Diagnosesoftware im Rahmen der Softwareprüfung wird in Abschnitt 3 genauer eingegangen.

2.1.2 Diagnoseaufgaben in der Produktion

Zu den wesentlichen Aufgaben der Diagnose innerhalb der Produktion gehört die Programmierung des Steuergeräts mit dem jeweils aktuellsten Softwarestand (Flashen) und die Kodierung der entsprechenden Variante. Bei der sogenannten Verbauprüfung wird die Steuergeräte-Identifikation am Bandende ausgelesen, um sicherzustellen, dass das richtige Steuergerät in das Fahrzeug eingebaut wurde. Um die Funktionalität am Bandende noch einmal sicherzustellen, erfolgt auch dort die Stellgliedansteuerung und es werden Messwerte aus dem Steuergerät ausgelesen. Schließlich erfolgen mithilfe der Diagnose eine letzte Kontrolle und ein Rücksetzen der im Steuergerät enthaltenen Fehlercodes. Dabei sollen während der Montage aufgetretene Fehler gelöscht und darüber hinaus sichergestellt werden, dass keine weiteren Störungen bestehen.

2.1.3 Diagnoseaufgaben im Service

Wichtigste Aufgabe der Diagnose im Service ist die Unterstützung der Werkstattmitarbeiter bei der Fehlersuche. Hierzu dienen ebenfalls die Verbauprüfung, Stellgliedtests, das Lesen von Steuergeräteeingängen und die Ansteuerung steuergeräteinterner Testroutinen. Außerdem kann mithilfe der Diagnose der Grad der Schädigung oder Abnutzung von bestimmten Bauteilen, wie z.B. von Sensoren festgestellt sowie die mögliche Restnutzungsdauer ermittelt werden.

Zur Identifikation von Fehlerursachen werden mittels der Diagnose die anliegenden Fehlercodes ermittelt. Ebenso findet eine erneute Fehlerüberprüfung nach Austausch oder Reparatur des defekten Bauteils statt. Auch die Flashprogrammierung und die Neukonfiguration des Fahrzeugs können in der Werkstatt vorgenommen werden.

2.2 Onboard- und Offboard-Diagnose

2.2.1 Onboard-Diagnose

Mit „Onboard-Diagnose“ werden die direkt im Steuergerät implementierten Diagnoseanteile bezeichnet. Sie dienen im Wesentlichen der Überwachung des einwandfreien Betriebs des Fahrzeugs bzw. der Identifikation von Störungen und Speicherung der entsprechenden Fehlercodes inkl. der Fehlerhistorie. Dazu gehört auch die Überwachung der korrekten Buskommunikation. Dies wird insgesamt als System- und Funktionsdiagnose bezeichnet. Eine besondere Aufmerksamkeit kommt dabei der Überwachung der abgasrelevanten Funktionen und Komponenten im Fahrzeug zu, weil gerade hier entsprechende gesetzliche Vorschriften in Form der OBD II (USA) und der EOBD (Europa) existieren.

Da zur Onboard-Diagnose auch die Bestimmung des Schädigungs- und Abnutzungszustands sowie die Ermittlung der möglichen Restnutzungsdauer von Fahrzeugkomponenten gehören, liefert sie auf diese Weise auch Informationen für die Wartung des Fahrzeugs.

2.2.2 Offboard-Diagnose

Die Offboard-Diagnose umfasst sämtliche Diagnoseanteile außerhalb des Fahrzeugs. Hierzu gehört die Darstellung der jeweiligen Diagnoseinformationen (Mensch-Maschine-Schnittstelle), wie z.B. die Anzeige der im Steuergerät gespeicherten Fehlercodes, die aktuelle Variantencodierung und die Darstellung ausgewählter Zustandsgrößen im Steuergerät. Speziell für den Einsatz in den Servicewerkstätten umfasst die Offboard-Diagnose auch Hinweise zur Identifikation möglicher Fehlerquellen und Reparaturanleitungen.

Die Art der Darstellung bzw. des Benutzerzugangs zur Diagnosefunktionalität ist stark auf die jeweilige Zielgruppe zugeschnitten. So werden in der Entwicklung komplette Diagnose-Werkzeugketten eingesetzt, die mittels sogenannter Autorensysteme auch die Erstellung der zugehörigen Diagnosedatenbasen (z.B. ODX) umfassen. In Produktion und Service werden dagegen funktional schlankere Benutzerzugänge in Form von Diagnosetestern eingesetzt. Speziell für den Servicebereich sind auch eine Reihe herstellerunabhängiger, sogenannter Scan-Tools verfügbar.

Die Verbindung zwischen dem Offboard-Diagnosesystem und dem Steuergerät erfolgt bislang über spezielle Diagnoseleitungen (K-Line) und heute überwiegend auch über den Fahrzeugbus, z.B. CAN. Der physikalische Zugang zur Onboard-Diagnose im Steuergerät wird dabei durch den sogenannten Diagnosestecker realisiert. Die Kommunikation zwischen dem Offboard-Diagnosetool oder -tester und der Onboard-Diagnose erfolgt über bestimmte Diagnoseprotokolle, wie z.B. KWP2000 (Keyword Protocol 2000 – ISO 14230) oder UDS (Unified Diagnostic Services – ISO 14229).

3 Entwicklungsbegleitendes Testen der Onboard-Diagnose

Der bereits erwähnte, hohe Anteil der Diagnose am Gesamtumfang der Steuergerätesoftware spiegelt sich in den Testkosten wieder. So umfasst das Testen der Diagnosefunktionalität ca. 25% der gesamten Testkosten eines Steuergeräts [2, 3].

3.1 Hardware-in-the-Loop-Simulation

Eine zentrale und wichtige Methode zum entwicklungsbegleitenden Testen von Steuergeräten bzw. Steuergeräteprototypen ist die Hardware-in-the-Loop- (HIL-) Simulation. Dabei wird das zu testende Steuergerät nicht im realen Fahrzeug verbaut, sondern an ein Simulationssystem angeschlossen, welches die Umgebung des Steuergeräts in Echtzeit simuliert. Damit sind umfassende und automatisierte Tests in einer simulierten Umgebung möglich.

Die HIL-Simulation kommt für unterschiedliche Tests zum Einsatz, z.B. für den Test von Einzelsteuergeräten, aber auch für den Test ganzer Steuergeräteverbunde. Dabei geht es zum einen um die korrekte Funktionalität. Zum anderen werden mit der HIL-Simulation auch das Kommunikationsverhalten der Steuergeräte am Bus sowie das Netzwerkmanagement getestet. Schließlich bildet die HIL-Simulation auch ein geeignetes Mittel zum Test des Steuergeräteverhaltens und der Steuergerätediagnose im Fehlerfall.

3.2 Automatisierter Test von Diagnosefunktionen

Abbildung 1 zeigt die Struktur eines HIL-Simulators, der mit allen für den Test der Steuergerätediagnose notwendigen Komponenten ausgerüstet ist.

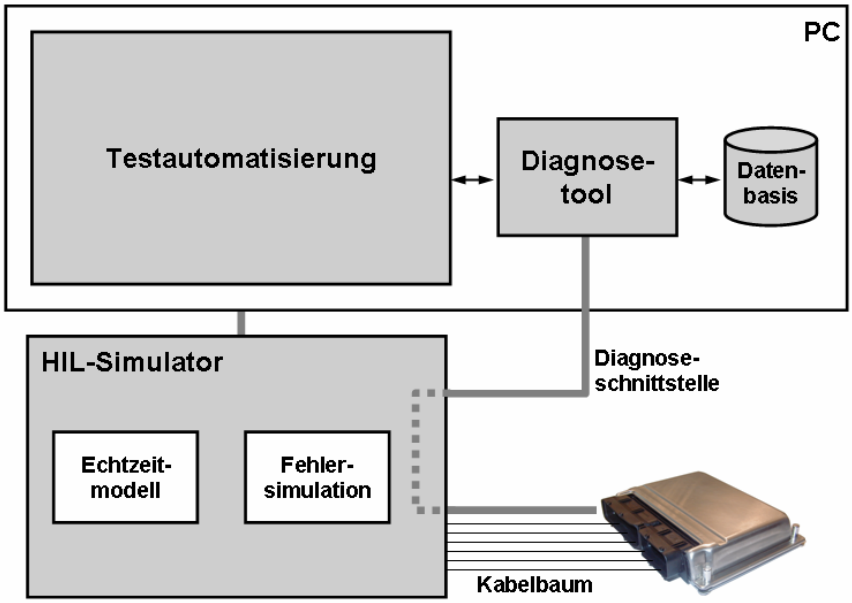

Abbildung 1: *Struktur eines HIL-Simulators mit integrierter Diagnose*

Zentrales Element ist das Testautomatisierungswerkzeug, in dem der komplette Testablauf in Form von Skripten oder grafischen Sequenzen hinterlegt ist. Das Testwerkzeug übernimmt in diesem Zusammenhang die Steuerung des gesamten Systems. Durch Zugriffe auf das Echtzeitmodell des HIL-Simulators sorgt es zunächst für das Anfahren des entsprechenden Betriebspunkts, in dem die Diagnosefunktion getestet werden soll. Dies ist notwendig, weil nicht alle Diagnosefunktionen in allen Betriebspunkten aktiv sind. Um die korrekte Erkennung elektrischer Fehler zu testen, wird mittels Relaiskarten in der elektrischen Fehlersimulation der zu detektierende Fehler erzeugt, beispielsweise ein Kurzschluss eines Steuergerätepins gegen die elektrische Fahrzeugmasse. Anschließend wird über das in den HIL-Simulator integrierte Diagnosetool der Fehlerspeicher des Steuergeräts ausgelesen. Dieser wird im Testwerkzeug mit dem für diesen Fehler erwarteten Fehlercode verglichen. Dieser Vorgang kann für alle Steuergerätepins und alle jeweils relevanten Fehlerarten automatisiert wiederholt werden. Anschließend werden die wichtigsten Testergebnisse in einem automatisch generierten Testreport dokumentiert.

Der beschriebene Testablauf stellt ein Beispiel für den Test der z.B. bei Motorsteuergeräten gesetzlich vorgeschrieben Onboard-Diagnose dar. Darüber hinaus umfasst die Diagnose auch herstellerspezifische Anteile, die ebenfalls mithilfe der HIL-Simulation getestet werden können.

4 Einsatz der Diagnose als Hilfsmittel für das Testen

Neben der Verwendung von HIL-Testsystemen für den Test der Onboard-Diagnose spielen Diagnose-Funktionen innerhalb eines HIL-Systems eine unterstützende Rolle als Hilfsmittel zum Testen.

4.1 Fehlerspeicher-Handling

Ein HIL-Testsystem wird immer dann benötigt, wenn der einfache, sogenannte “Brett-Aufbau“ des Steuergerätes und der Aktoren und Sensoren sowie die signaltechnische Beschaltung über Stimuli funktional nicht ausreichend ist. Plausible Signalverläufe der Eingangsgrößen eines Steuergerätes sind notwendig, um das Steuergerät im “normalen Fahrmodus“ zu betreiben. Diese Zustände lassen sich nur durch entsprechende Umgebungsmodelle simulationstechnisch berechnen.

Bei der Inbetriebnahme eines HIL-Testsystems, wenn sukzessive alle Eingangs- und Ausgangsgrößen des Steuergerätes mit den I/O-Schnittstellen des Echtzeitrechners einschließlich eventuell benötigter Restbussimulationen in Betrieb genommen werden, muss permanent überprüft werden, ob das Steuergerät ohne Fehlerspeicher-Einträge betrieben werden kann. Hierzu ist das gezielte Zurücksetzen und Auslesen einzelner oder aller Fehlerspeichereinträge notwendig. Des Weiteren ist es essentiell, die Plausibilität der anliegenden Signalgrößen mit den Werten aus den Umgebungsdaten der DTCs[1] abzugleichen.

Diese Funktionalität wird natürlich auch während des manuellen oder automatisierten Testbetriebs eines HIL-Simulators verwendet. Dem Einsatz während der Inbetriebnahmephase kommt jedoch eine besondere Bedeutung zu, da durch die Fähigkeit zum fehlerfreien Betrieb eines Steuergeräts die Referenz des Testsystems definiert wird.

4.2 Stellgliedansteuerung

Die Möglichkeit, bestimmte Ausgänge des Steuergerätes (Stellglieder, Aktoren) direkt über entsprechende Diagnose-Services ansteuern zu können, kann sowohl in einer Inbetriebnahmephase als auch im geregelten Testbetrieb sinnvoll verwendet werden.

Wird das Steuergerät im geschlossenen Regelkreis betrieben, ist das explizite Ansteuern von definierten Zuständen von Aktoren nur durch entsprechende Beschaltung von Eingängen und unter Kenntnis der Signalpfade in der Steuergerätefunktionalität möglich. Es müssen dann also entsprechende Fahr- oder Betriebsmanöver simulationstechnisch vorgegeben werden. Der Zugriff über die entsprechenden Diagnoseservices schneidet diese Signalpfade quasi auf und steuert die Ausgänge direkt an.

4.3 Variantenkodierung

Dem Trend, Steuergeräte in zunehmendem Maße generischer und damit kostengünstiger ausprägen zu müssen, aber auch variantenspezifische Charakteristika zu ermöglichen, muss auch das Testsystem Rechnung tragen. HIL-Systeme werden demzufolge so konzipiert, das sich nicht nur ein konkretes Steuergerät anschließen und testen lässt, sondern auch Varianten des Steuergeräts.

---

[1] DTC – Diagnostic Trouble Code = Fehlerspeichereintrag mit Fehlerbezeichnung, Fehlercode, Umgebungsbedingungen zum Zeitpunkt des Auftretens des Fehlers usw..

Je nach Verwendungsart des Simulators muss das angeschlossene Steuergerät daher für den jeweiligen Einsatzzweck variantenkodiert werden. Bei neu verbauten Sensoren oder Aktoren erfolgt in vielen Fällen zunächst ein „Anlernen“, d.h. Ihr spezifisches Mess- oder Ansteuerverhalten muss dem Steuergerät erst „bekanntgemacht“ werden. Hierzu sind besondere Funktionalitäten der Diagnosegeräte anzusprechen, wie sie ansonsten auch direkt im Werkstattbetrieb durch das Service-Personal verwendet werden.

Neben der Möglichkeit die benötigte Variante zu kodieren, muss in umgekehrter Richtung die momentan aktive Kodierung auch ausgelesen werden können, um z.B. im Testreport protokolliert zu werden oder um spezielle Umfänge innerhalb einer komplexen Test-Suite festzulegen bzw. vorauszuwählen.

Darüber hinaus kann es erforderlich sein, in Abhängigkeit der Variantenkodierung auch unterschiedliche Echtzeitmodelle zu verwenden und diese dann entsprechend auf den Simulator zu laden.

4.4 Zugriff auf steuergeräteinterne Verstell- und Messgrößen

Der Zugriff auf steuergeräteinterne Verstell- und Messgrößen in einem HIL-Szenario erfolgt üblicherweise über Kalibrierschnittstellen und eine Fernbedienung der betreffenden Tools, z.B. über ASAP3 oder ASAM MCD-3MC (M = „Measurement“, C = „Calibration“).

Nicht immer kann jedoch auf diese Infrastruktur zurückgegriffen werden. Sei es, dass in späten Entwicklungsphasen diese Schnittstellen oder die betreffenden Beschreibungsdaten nicht zur Verfügung stehen oder dass der Zugriff auf diese Größen ausschließlich über die Diagnose-Funktionalität gegeben ist, da diese auch für den Service relevant und damit auch im Serienstand des Steuergeräts zugänglich bleiben.

Alle Steuergeräte bieten heute die Möglichkeit bestimmte Messgrößen oder ganze Messwert-Blöcke pollend, d.h. durch wiederholtes Auslesen der entsprechenden Größen aus dem Steuergerät zu erfassen. In zunehmendem Maße, z.B. bei Verwendung des UDS-Protokolls, werden jedoch auch höherwertige Mess-Services implementiert.

5 Stand der Technik bzgl. der dSPACE Tools

5.1 Anbindung proprietärer Diagnoseschnittstellen

Die Vielzahl unterschiedlicher Diagnosetools, -interfaces, -protokolle und -datenbeschreibungen, die in den vergangenen Jahren bei den verschiedenen OEMs und Zulieferern eingesetzt wurden, stellt einen herstellerunabhängigen HIL-Testsystemanbieter vor eine große Herausforderung. Abbildung 2 zeigt einen Auszug aus dem Umfang der implementierten Diagnoseanbindungen im Rahmen von HIL-Testsystemen bei dSPACE aus dem Jahr 2000. Je nach Ausprägung waren dies reine Hardware-API-Anbindungen, Lösungen auf Protokollebene oder bereits Fernbedienungen von Diagnose-Tools mit Datenbasis.

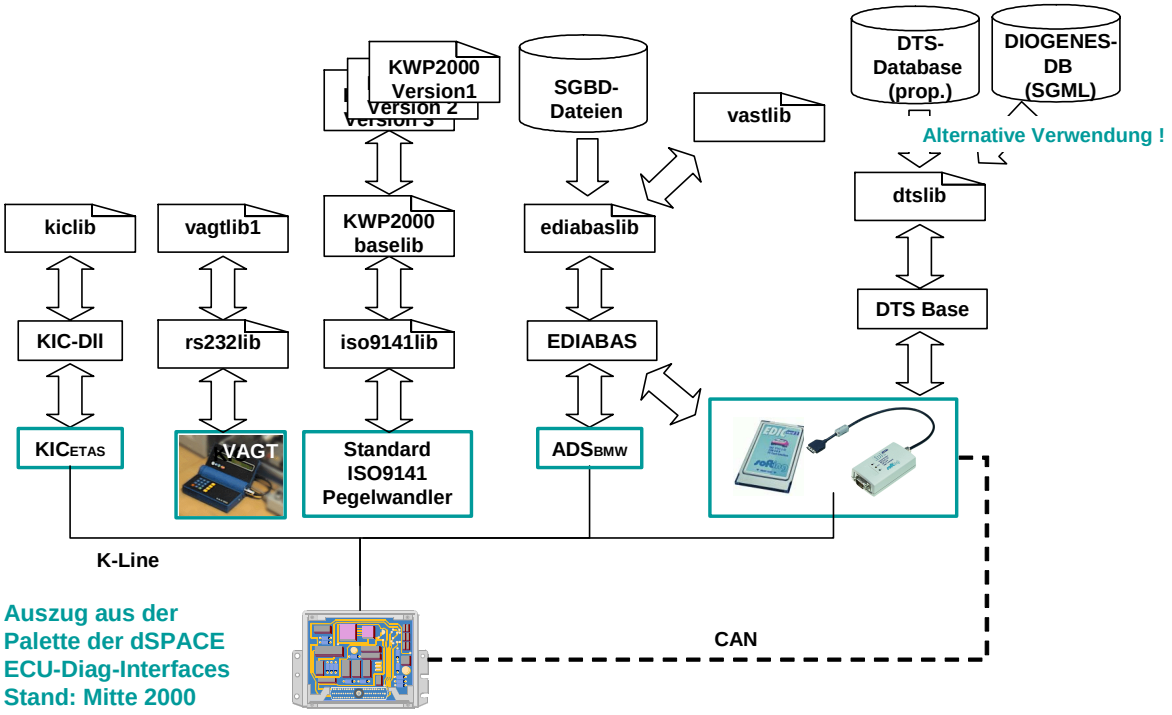

Abbildung 2: *Kundenspezifische Diagnose-Lösungen (Stand 2000)*

Die Einbindung und Nutzung der Diagnose in der HIL-Anwendung erfolgte entweder durch die direkte Implementierung des entsprechenden Diagnoseprotokolls innerhalb der Testautomatisierungssoftware oder durch eine Fernsteuerung des entsprechenden Diagnosetools aus dem HIL-System heraus. Dabei ist nicht nur der Umfang der unterstützten Fernsteuerfunktionen von Diagnosetool zu Diagnosetool unterschiedlich. Die Art und Weise, wie die Einbindung erfolgt, hängt auch sehr stark von der verwendeten Schnittstellentechnologie ab. Beispielsweise wird der „klassische“ VAG-Tester über die serielle PC-Schnittstelle ferngesteuert, andere Tools bieten DDE- (z.B. DiagRA von RAConsulting) oder COM-Schnittstellen (z.B. DTS von Softing ab Version 7 oder CalDesk von dSPACE). Es wird deutlich, dass ein sehr hoher Entwicklungs- und Pflegeaufwand für diese Vielzahl technologisch unterschiedlicher Module bestand und teilweise bis zum heutigen Tag besteht. Der Wiederverwendungswert über Projekte hinweg war nur gering und die kundenspezifischen Engineeringanteile entsprechend groß.

Um diese Schnittstellenvielfalt einzudämmen, hat es in den vergangenen Jahren diverse Standardisierungsaktivitäten gegeben. Das Ergebnis ist der vom ASAM e.V. definierte Standard ASAM MCD-3D (ISO22900-3). Dieser Standard definiert vor allem ein umfangreiches API (Application Programming Interface), also eine Programmierschnittstelle in Form eines Objektmodells mit ca. 800 verschiedenen Methoden, das u.a. zur Automatisierung von Diagnosetools in HIL- oder auch Prüfstandszenarien dient. Durch die Verwendung der UML (www.uml.org) ist der Standard von der verwendeten Schnittstellentechnologie und Programmiersprache unabhängig. Es existieren jedoch konkrete Referenzimplementierungen für Java, COM/DCOM und C++.

Sowohl der ASAM MCD-3D-Standard, aber auch die ebenfalls unter dem Dach des ASAM e.V. bzw. in der ISO betriebenen Standardisierungen zum Diagnosedatenfor-

mat (ASAM MCD-2D (ODX)/ ISO22900-1) und zur Anbindung der Diagnosehardware (ISO22900-2) wurden sukzessive in der dSPACE Toolwelt implementiert.

5.2 Standardisierte Schnittstellen mit CalDesk

Das Mess- und Applikationstool CalDesk, das ursprünglich für den Mess- und Verstellzugriff auf Steuergeräte ausgelegt war, wurde um die integrierte Steuergerätediagnose erweitert. Wurden bislang für die MC-Funktionalitäten die bekannten ASAM Standards ASAM MCD 1 (CCP und XCP), ASAM MCD 2MC (ASAP2), ASAP3 und ASAM MCD 3MC unterstützt, kamen im Zuge der funktionalen Erweiterung auf den Diagnosebereich auch die „ASAM D-Standards“ hinzu.

Über die standardisierte Serverschnittselle ASAM MCD-3D wurde ein sogenanntes Diagnose-Laufzeitsystem angebunden. Hierbei bietet dSPACE ein integriertes System an, das ODX-Daten einliest und mittels einer für Mess- und Kalibrieraufgaben verwendeten Schnittstellenhardware, z.B. ein CAN-Interface, den Diagnosezugriff ermöglicht (Abbildung 3). Das hierbei von dSPACE integrierte Diagnose-Laufzeitsystem stammt von der Firma IN2Soft.

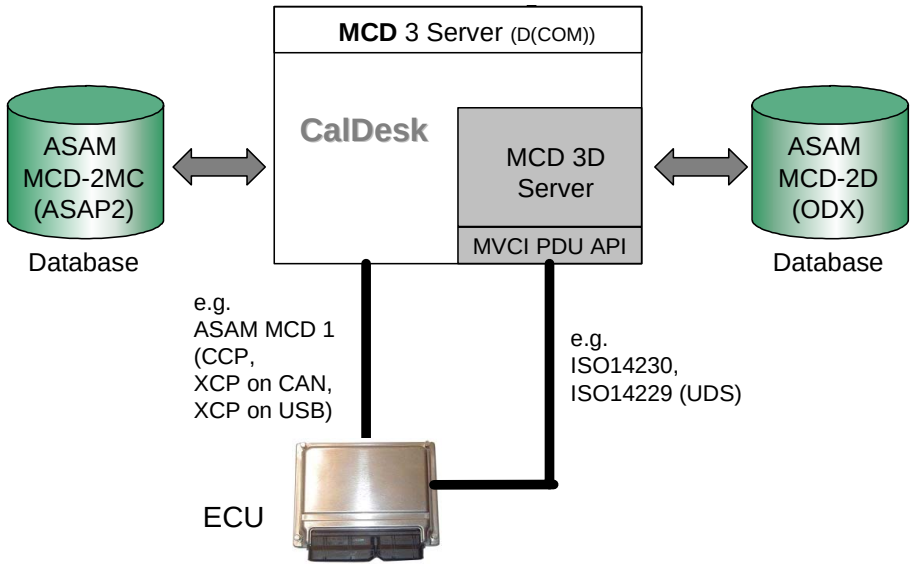

Abbildung 3: *CalDesk mit integrierter Diagnosefunktionalität*

Die grafischen Bedienelemente z.B. zum Zugriff auf den Fehlerspeicher oder zur Aktivierung von beliebigen Diagnose-Services sind hierbei direkt in die Bedienfenster zum Verstellen oder Messen von Steuergerätevariablen integriert. Diagnoseereignisse, wie etwa die Veränderung der Anzahl der eingetragenen Fehlerspeichereinträge können direkt zeitlich korreliert werden zu den Messsignalen über das XCP-Interface.

CalDesk legt neben diesen interaktiv zu nutzenden Funktionalitäten auch die standardisierte Programmierschnittstelle ASAM MCD 3D in der Technologie COM/DCOM frei und ermöglicht hierdurch automatisierte Zugriffe durch ein externes Client- bzw. Automatisierungssystem.

5.3 Nutzung standardisierter Schnittstellen in der HIL-Testautomatisierung

Die MCD-3-Serverschnittstelle wiederum kann im Rahmen von Automatisierungsszenarien am realen Prüfstand oder am HIL-Simulator genutzt werden, indem von einem entsprechenden Automatisierungssystem darauf zugegriffen wird. Ein solches Automatisierungssystem kann z.B. das dSPACE Testautomatisierungstool AutomationDesk sein, mit Hilfe dessen u.a. dSPACE HIL-Simulatoren einschließlich deren Fehlersimulations-Hardware automatisiert betrieben werden können (Abbildung 4). In grafischer Form können hier u.a. Diagnoseabläufe als Bestandteile hierarchischer Testsequenzen programmiert werden. Neben der CalDesk-Integration existieren auch Anbindungen an Diagnosetools anderer Hersteller, wie etwa die der Firmen IN2SOFT, Softing, T-Systems oder an Applikationstools wie INCA und CANape. Die dunklen Pfeile verdeutlichen hierbei die standardisierten Interfaces, während die weißen Pfeile proprietäre Schnittstellen darstellen.

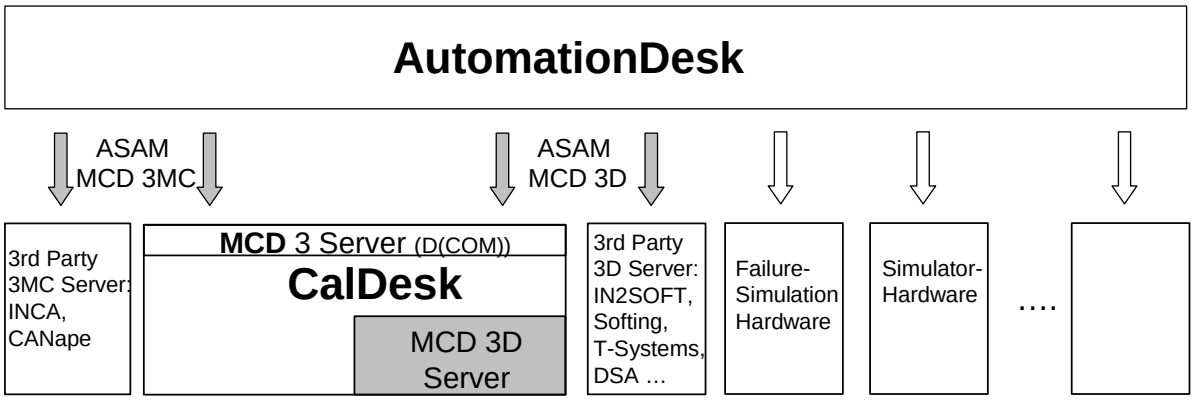

Abbildung 4: *AutomationDesk Schnittstellen*

Der nachfolgend dargestellte AutomationDesk Screenshot (Abbildung 5) verdeutlicht eine einfache, sequentielle Ablauffolge von elementaren Diagnose-Serviceaufrufen. In diesem Beispiel wurde das Interface zum Softing DTS7 System verwendet. Die zur Verfügung stehenden Basisblöcke bilden hierbei direkt MCD 3D-Befehle ab, können allerdings auch zu komplexeren Sequenzen kaskadiert und damit wiederverwendbar gemacht werden.

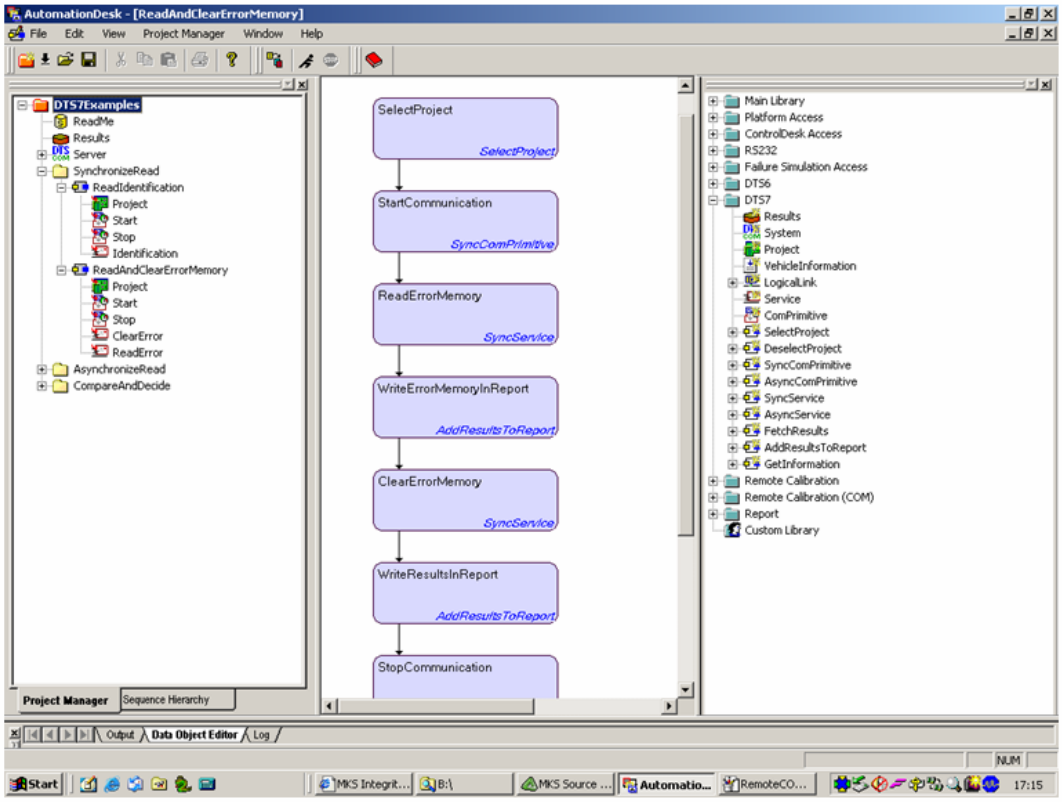

Abbildung 5: *AutomationDesk Screenshot*

6 Praktische Probleme

Wie in den vorherigen Abschnitten gezeigt wurde, stellt die integrierte Diagnose ein zentrales Element von HIL-Simulatoren dar. Tatsächlich werden heute kaum HIL-Simulatoren eingesetzt, an die kein Diagnosesystem angebunden oder integriert ist. Trotzdem oder gerade deswegen stellt die Integration der Diagnose in HIL-Testsysteme die Beteiligten immer wieder vor eine Reihe von praktischen Problemen.

6.1 Schwachstelle Diagnosedaten

Da Diagnosewerkzeuge und –tester zunehmend generisch, die Diagnosefunktionalität jedoch vielfach steuergerätespezifisch ist, gehört zu einem Steuergerät jeweils eine Beschreibung, welche Diagnosefunktionalität (Dienste) im Steuergerät vorhanden sind. Diese Information wird beispielsweise von einem Diagnosewerkzeug ausgewertet, um dem Benutzer die entsprechenden Diagnosedienste anzubieten.

Der Einsatz der Diagnose im Rahmen von HIL-Simulatoren und auch der Test der Diagnosefunktionalität setzen daher eine Beschreibung der implementierten oder zu implementierenden Diagnosedienste voraus. In der Vergangenheit wurden diese Informationen in sogenannten Steuergerätebeschreibungsdateien, kurz „SGBD“ abgelegt. Die SGBD-Formate waren weitestgehend durch das verwendete Diagnosewerkzeug festgelegt und somit proprietär. Folglich existierten eine ganze Reihe unterschiedlicher, proprietärer Formate zur Beschreibung von Diagnosediensten, wie z.B. EDIABAS-SGBD, Diagnostic-Tool-Set-Datenbasen und DIOGENES-Datenbasen (in SGML).

Ein Lösungsansatz, der in den letzten Jahren verfolgt wurde, liegt in der Standardisierung der Diagnosebeschreibung. Dazu wurde das ODX-Format entwickelt. ODX steht dabei für „Open Diagnostic Data Exchange“ und wurde im Rahmen des ASAM e.V.und in Kooperation mit der ISO unter der Bezeichnung „ASAM MCD-2D (ODX)“ standardisiert. Ziel von ODX ist es, ein einheitliches Datenformat zur Beschreibung von Diagnoseinformationen und somit zur Parametrierung von Diagnosetools und –testern bereitzustellen, das einen prozessweiten Austausch der Diagnosedaten und somit eine effizientere Diagnoseentwicklung ermöglicht.

Der allgemeine Anspruch von ODX, die Anforderungen der vielen an der Standardisierung beteiligten Unternehmen, insb. der OEMs, abzudecken, hat dazu geführt, dass das ODX-Format selber sehr komplex ist. Die Erstellung einer ODX-Beschreibung für ein Steuergerät erfordert somit neben dem entsprechenden Autorensystem (Erstellungswerkzeug) sehr viel ODX-Expertenwissen und verschiedene Regelwerke (Autorenrichtlinien) zur Beherrschung dieser Komplexität. Trotzdem bedeutet die Verwendung von ODX nicht per se Austauschbarkeit. Dies liegt beispielsweise daran, dass unterschiedliche OEMs unterschiedliche Teile von ODX nutzen und sich die damit erstellten Diagnosebeschreibungen nicht direkt wiederverwenden lassen.

In der Praxis ist noch ein entscheidender Aspekt bei der Einführung von ODX zu berücksichtigen: die Migration der heutigen Toolwelten zur Erstellung der Diagnosedaten und der heutigen Laufzeitsysteme hin zu einer ASAM konformen Ausrichtung. In diesem Zusammenhang werden sehr unterschiedliche Ansätze bei den OEMs verfolgt. Ein solcher Ansatz kann in der schrittweisen Einführung von ODX bestehen, wobei z.B. ODX-Dateien in Altformate konvertiert werden, um zunächst die etablierte Laufzeit-Toolwelt stabil halten zu können [4].

6.2 Schwachstelle Verfügbarkeit

Ein weiteres Problem im Zusammenhang mit Diagnose und HIL-Simulation ist die zeitliche Verfügbarkeit von Diagnosedaten und Diagnosesoftware. Der Test oder die Verwendung von Diagnosefunktionalität im HIL erfordert es, dass die entsprechenden Steuergeräte die notwendige oder zu testende Diagnosefunktionalität überhaupt implementieren und dass die entsprechenden Diagnosebeschreibungen vorliegen. Da der HIL-Test immer früher im Entwicklungsprozess erfolgt und permanent sowie entwicklungsbegleitend, z.B. für Regressionstests neuer Softwarestände, eingesetzt wird, um mögliche Fehler bereits so früh wie möglich aufzudecken, stellt sich zunehmend das Problem, dass zu diesem Zeitpunkt noch gar keine Diagnosedaten vorhanden sind, oder die an den HIL-Simulator angeschlossenen Steuergeräte noch gar keine Diagnosefunktionalität unterstützen. Die Folge: eine späte Implementierung der Diagnosefunktionalität innerhalb des Entwicklungsprozesses macht das Testen vielfach erst spät möglich, die Fehlerbehebung aufwändiger und die Entwicklungszyklen länger als notwendig.

6.3 Schwachstelle Automatisierungsschnittstelle

Auch bei der Standardisierung der API im Rahmen von ASAM MCD-3D (s. Kapitel 5.1) gab es – ähnlich wie bei ODX – eine Vielzahl an spezifischen Änderungswünschen und Erweiterungen, um die vielen unterschiedlichen Anforderungen, insbesondere der an der Standardisierung beteiligten OEMs, abzudecken. Dies hat zur

Folge, dass die Schnittstelle äußerst komplex ist und ihre Anwendung dediziertes Expertenwissen erfordert. Darüber hinaus ist der Standard noch relativ jung und somit noch relativ häufigen Änderungen unterworfen. Die Unterstützung des Standards durch ein Tool bedeutet daher nicht automatisch, dass das Tool gegen ein anderes, den Standard unterstützendes Tool einfach ausgetauscht werden kann.

Das ASAM-MCD-3D-API bietet funktional einen direkten Zugang zur Diagnosefunktionalität. Für die direkte Verwendung im Sinne einer Automatisierungs- und Testaufgabe ist diese Zugangsebene jedoch nur bedingt geeignet. Typische Diagnoseaufgaben im Rahmen einer HIL-Anwendung, wie z.B. in Abschnitt 3 beschrieben, setzen sich i.d.R. aus einer Reihe von einzelnen Diagnosezugriffen zusammen. Diese erfordern überdies eine relativ komplexe Parametrierung in Form von speziellen Objekten, die das Diagnosesystem („Server“), das Diagnoseprojekt, das Fahrzeug („Vehicle Information Table“), die relevanten Steuergeräte („Locigal Links“) und die eigentlichen Zugriffsfunktionen („COM Primitives“ und „Services“) beschreiben.

In der HIL-Praxis taucht daher immer wieder die Situation auf, dass ein HIL-Anwender, der die Diagnose nur als Black-Box betrachtet, mit der direkten Diagnoseanbindung auf Ebene des MCD-3D-Standards innerhalb des HIL-Systems und bei der Testerstellung nichts anfangen kann. Der Anwender erwartet vielmehr höherwertige, aggregierte Diagnosefunktionen, etwa auf dem Zugangslevel, den er von einem Standard-Diagnosetester kennt. Dafür muss dann auch die Parametrierung des Diagnosetools entsprechend vorbereitet sein.

Abschließend lässt sich beobachten, dass ein OEM sich in der Regel trotz Standardisierung auf einen konkreten Diagnosetool-Anbieter konzentriert und beschränkt, weil er nur so die Komplexität der Diagnose in Zusammenarbeit mit diesem Partner, der die spezifischen Anforderungen und Probleme kennt, in den Griff bekommt. Dies hat letztlich aber zur Folge, dass kaum „natürliche Cross-Tests“ stattfinden, weil die Implementierung eines Standards wiederum OEM-spezifisch ist und nicht durch die Verwendung über mehrere OEMs hinweg erprobt wird, folglich nicht „auf natürliche Weise“ reifen kann.

6.4 Firmenspezifische Standards zur „Verkryptung“

ODX als klartextlesliches Format kann die Sicherheitsanforderungen in bestimmten Prozessschritten bei einer verteilten Entwicklungs- und Zuliefererlandschaft prinzipbedingt nicht erfüllen. Unter anderem aus diesem Grund wurden OEM-spezifische Binärformate eingeführt, die direkt von den dort verwendeten Laufzeitsystemen verarbeitet werden können. Hierdurch entstehen wiederum proprietäre Schnittstellen, die bei Weitergabe über Firmen- bzw. Toolgrenzen hinweg eine bidirektionale Konvertierung erfordern, was üblicherweise nicht ohne Informationsverlust bzw. nicht ohne Zusatzaufwand erfolgen kann.

6.5 Know-How-Hürde zwischen HIL-/Test-Welt und Diagnose-Welt

Die obigen Ausführungen zeigen, dass der Einsatz und die Verwendung der Steuergerätediagnose aufgrund der hohen technischen Komplexität einerseits und der engen Verzahnung mit Prozessfragen von der Entwicklung über die Produktion bis hin zum Service andererseits eine große Herausforderung darstellt. Dies lässt sich auch daran erkennen, dass sich gerade im Umfeld Diagnose immer mehr Unternehmen

positionieren. Dies umfasst zunächst natürlich die Anbieter von Diagnosetools. Darüber hinaus entstehen aber immer mehr Angebote im Bereich des Consultings und der Prozessberatung.

Solche Ansätze sind geeignet, um entwicklungs- oder unternehmensweite Experten, Vorgehensweisen und „Sub-Standards“ zu etablieren und somit die Komplexität des Themas entlang der „üblichen Prozesskette“ zu beherrschen. Die im Rahmen der HIL-Simulation aufkommenden Probleme werden dadurch aber vorwiegend nicht adressiert. Auch und gerade die HIL-Technologie erfordert sehr viel Expertenwissen, z.B. im Bereich der Modellierung und der Testautomatisierung. Da sich die Diagnose aus Sicht der HIL-Technologie jedoch weitgehend auf eine Schnittstellenfrage reduzieren lässt, existieren i.d.R. weder ein Diagnose-Experte im direkten Umfeld der HIL-Technologien, noch ein Bewusstsein für die Notwendigkeit der Verankerung von entsprechendem Diagnosewissen. Die Verantwortung für die Bereitstellung und das Vorhandensein der richtigen Diagnosedaten und der Diagnosesoftware wird auf die eigentlichen Diagnoseexperten oder die Steuergeräteverantwortlichen projeziert. Somit existieren zwischen der Diagnose- und der HIL-Technologie sowohl eine Know-How-Barriere als auch eine Zuständigkeitshürde.

7 Lösungsansätze

7.1 Praxistaugliche Umsetzung der Standardisierung

Die Darstellung der praktischen Probleme in Kapitel 6 darf nicht den Eindruck erwecken, als ob Standardisierung einen Irrweg darstellte. Ganz im Gegenteil: vorhandene Standards müssen verwendet und in Produkten implementiert werden.

Allerdings muss ein Umfeld über Anbieter- und Anwendergrenzen hinweg geschaffen werden, dass zu einer wirklichen Konformität und Praxis-Tauglichkeit der standardisierten Schnittstellen in den Produktimplementierungen führt. Nichts ist kontraproduktiver, als mit hohem Entwicklungsaufwand entwickelte und implementierte Standards, die dann aufgrund von spezifischen Erweiterungen, Inkonformitäten, grundsätzlich fehlender Funktionalität oder unzureichender Performance nicht in die Anwendung kommen.

Ein probates Mittel um diesen Problemen frühzeitig zu begegnen, sind öffentliche Cross-Tests über Firmengrenzen hinweg. Diese Tests müssen wiederholt durchgeführt werden. Dies erfordert, dass sich die verschiedenen Toolhersteller an diesen Tests beteiligen, und dass die OEMs die Infrastruktur für derartige Tests in Form von Steuergeräten und Daten bereitstellen.

Der ASAM e.V. hat 2006 einen zweiten erfolgversprechenden Weg beschritten: das Bereitstellen von offiziellen Format-Checker-Tools für die Datenbasis-Standards. Wünschenswert wäre hier eine Adaption der Endanwender, d.h. der OEMs, in der Form, dass in den Ausschreibungen für zukünftige Steuergeräteprojekte gefordert wird, die Konformität der betreffenden Dateien in der kompletten Prozesskette ständig gegen diese offiziellen Checker zu gewährleisten.

7.2 Convenience Layer

In den Kapiteln 6.3 und 6.4 wurde dargestellt, dass die vorhandenen standardisierten API-Schnittstellen nicht ideal ausgerichtet sind für Anwender, die aus dem Blickwinkel Steuergerätetest bzw. HIL-Simulation das Themenfeld Diagnose oder Steuergeräte-Applikation berühren.

Aus diesem Grund erscheint es sinnvoll, hier eine Art „Convenience-Layer“ einzuführen, der für typische Diagnose- und Applikationsabläufe elementare MCD3-API-Befehlsketten aggregiert und vorparametriert. Diskussionsansätze zu dieser Themenstellung werden immer wieder laut, nur sind bislang noch keine verwertbaren Arbeitsergebnisse bekannt geworden, die den Anspruch hätten, mehr als eine firmenspezifische Lösung darzustellen.

7.3 Gesamtarchitektur zur Einbindung der Steuergeräte-Diagnose in HIL-Testsysteme

In Abbildung 6 wird gezeigt, wie der Convenience Layer für Diagnose in eine Gesamtarchitektur für HIL-Testsysteme eingebettet werden kann. Auf der untersten Ebene finden sich die bekannten funktionalen Komponenten eines HIL-Testsystems. Capture-Services zur Echtzeitdatenerfassung, Stimulus-Services zur Echtzeit-Signalvorgabe und Fehlermodelle laufen parallel zum eigentlichen Echtzeitmodell auf dem Echtzeitrechner. Der Zugriff aus der Testautomatisierungssoftware erfolgt über ein entsprechendes Echtzeitrechner-Interface. Die Ansteuerung der Fehlersimulations-Hardware zur Erzeugung elektrischer Fehler erfolgt über eine weitere Schnittstelle, hier als FIU-API bezeichnet. Beide Schnittstellen sind heute proprietär. Daneben erfolgt die Integration von Diagnose- und auch Kalibriertools, wie in den vorangegangen Abschnitten beschrieben, heute i.d.R. über den ASAM-MCD-3-Standard, wobei die Anbindung i.d.R. getrennt nach MC- und D-Anteil erfolgt.

Der zuvor beschriebene Convenience-Layer würde somit eine Funktionsschicht oberhalb dieses Standards bilden. Für die Anwendung in HIL-Testsystemen könnte der Convenience-Layer so ausgedehnt werden, dass er ebenfalls die proprietären Schnittstellen des HIL-Testsystems in Form eines HIL-Plattform-APIs abstrahiert. Dies würde eine entsprechende Standardisierung der HIL-Plattform-API erfordern bzw. mit einschließen.

Auf der obersten Ebene ist es nun möglich anwendungs- und domänenspezifische Testbibliotheken zu definieren und ggfs. zu standardisieren. Dazu gehören z.B. spezielle Testbibliotheken für OBD-Tests und Fahrzyklen im Motor- und Antriebsstrangbereich. Genauso könnten Testbibliotheken für andere Domänen, wie z.B. Komfortbereich und Infotainment definiert werden.

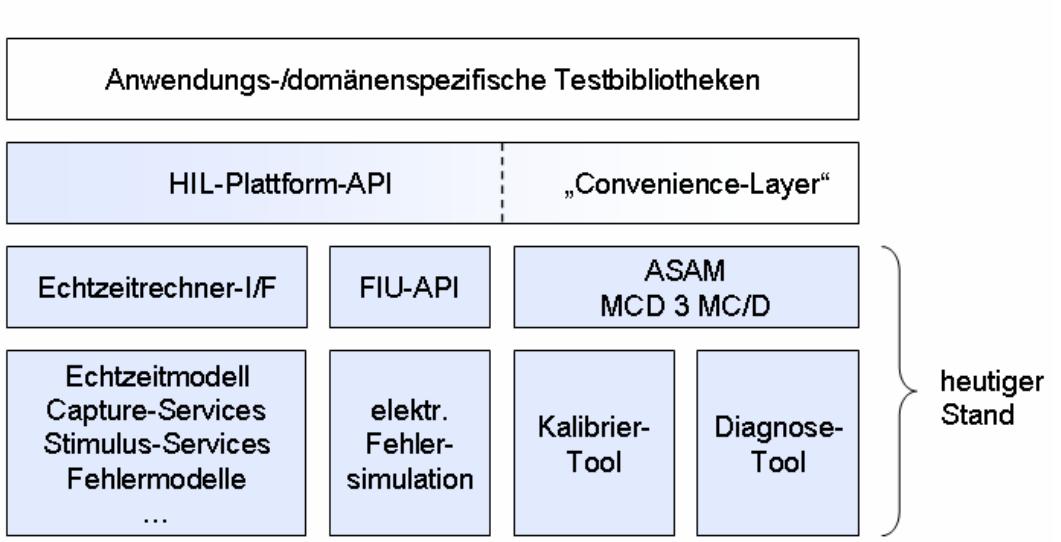

Abbildung 6: *HIL-Gesamtarchitektur*

Für die Standardisierung eines HIL-Plattform-APIs laufen – initiiert durch dSPACE – seit Herbst 2005 bereits erste Aktivitäten, an denen sowohl OEMs und Zulieferer, wie auch Toolanbieter beteiligt sind. Ziel ist die Ausarbeitung eines Projektvorschlags, der 2007 in den ASAM eingebracht werden soll. Die dabei stattfindenen Überlegungen wurden ursprünglich aus einer anderen Motivation als der hier beschriebenen Diagnoseproblematik heraus ausgelöst. Die Ausführungen dieses Beitrags zeigen jedoch, dass eine Verknüpfung der Themen, insb. wegen der großen Bedeutung der Diagnose für das Testen mithilfe der HIL-Simulation, sinnvoll ist.

8 Literatur

[1] KLEE, P.; KNIRSCH, M.; WILLIMOWSKI, M.: Herausforderungen der Diagnoseentwicklung in der Motorsteuerung, in: Onboard-Diagnose – Status der Gesetzgebung und Auswirkungen auf die Fahrzeugentwicklung, expert-Verlag, Renningen, 2005

[2] VARCHMIN, U.: Diagnose von E/E-Systemen im Automobil, EUROFORUM, 2005

[3] WEINFURTHER, J.: Automotive Diagnostic and Security System Issues, Vector Congress, 2005

[4] KAISER, D.; GRZONDZIEL, J.; WEYRATH, T.; HÜMPFNER, B.: ODX-Best2-Converter, ASAM User Days, 2006