Wiederverwendbarkeit und Management von modellbasierten X-in-the-Loop Tests mit TTCN-3 Embedded

Jürgen Großmann[1], Hans-Werner Wiesbrock[2]
[1] FOKUS, Fraunhofer Institut, Kaiserin-Augusta-Allee 31,10589 Berlin,
juergen.grossmann@fokus.fraunhofer.de
[2]IT Power Consultants, Gustav-Meyer-Allee 25, 13355 Berlin,
hans-werner.wiesbrock@itpower.de

Motivation und Einleitung

Der Einzug moderner Software Technologie und eingebetteter Systeme in unseren Alltag ist überall zu spüren, in den heutigen Haushaltsgeräten wie im neuen Automobil. Anstelle der alten Dampfregler oder Röhren steuern heute kleinste integrierte Schaltungen die Geschwindigkeiten und Heizströme. Betrachten wir beispielhaft die Ansteuerung einer Drosselklappe im Automobil.

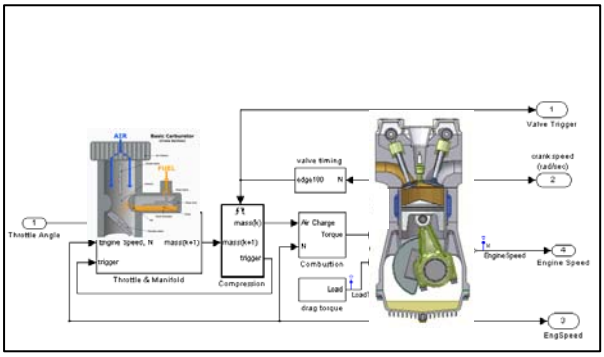

Abbildung 1: Drosselklappenansteuerung im Motor

Je nach Öffnung der Drosselklappe wird mehr oder weniger Luft der Verbrennung zugeführt und entsprechend das Drehmoment des Motors geregelt. Ziel ist es, mit möglichst geringem Benzinverbrauch die vom Fahrer gewünschte Geschwindigkeit zu erreichen. Das ist mit einer konstanten Einstellung nicht möglich. Deshalb übernimmt ein Motorsteuergerät diese Aufgabe. Dieses öffnet und schließt die Drosselklappe je nach gewünschter Geschwindigkeit und Beschleunigung, abhängig von verschiedenen anderen Parametern unseres Motors. Die für die Steuerung verwendeten Algorithmen werden heutzutage in Software realisiert.

Wenn wir ein solches System testen wollen, reicht es nicht, spezielle Eingaben in das System zu geben und auf die korrekten Antworten zu horchen, zu komplex ist das Wechselspiel von Testobjekt und Umgebung. Die Ansteuerung der Drosselklappe durch die Schaltung wirkt auf den Motorlauf, die Geschwindigkeit und das Drehmoment ändern sich. Damit ist aber auch der Öffnungswinkel zu modifizieren. Unser System ist eine Regelung mit Rückkopplung. Können Steuerungen, Systeme ohne Rückkopplung, noch durch Open-Loop-Tests vollständig dynamisch überprüft werden, so ist dies jedoch für Regelungen mit Feedback im Allgemeinen nicht möglich. Wir haben in geeigneter Weise die Auswirkungen unserer Ansteuerung auf die Umgebung für die weitere Testeingabe zu berücksichtigen. Gerade dies zeichnet einen Closed-Loop-Test aus.

Um die Auswirkungen auf die Umgebung und damit auch den Rückfluss der Ansteuerung durch das Testobjekt zu erfassen, wird für den Test ein Modell der Umgebung entwickelt. Testen von Regelungen benötigt somit nicht nur das Testobjekt und einen Testrahmen, sondern auch ein geeignetes Modell der späteren Umgebung, in die es eingebettet werden soll und die es regeln soll.

Andererseits dienen Tests dem Nachweis korrekter Funktionalität. So verlangen verschiedene Qualitätsstandards die Überprüfung von Anforderungen durch diverse Tests und wir benötigen eine systematische Testverwaltung mit Tracing, um dieser Forderung zu genügen. Das ist nicht effektiv möglich, wenn wir für jeden Test ein eigenes Umgebungsmodell entwickeln wollten. Deshalb wird hier eine allgemeine Architektur von Closed-Loop-Tests vorgeschlagen, die eine systematische Verwaltung aber auch Wiederverwendung von Tests in späteren Entwicklungsphasen und projektübergreifend erlaubt. Sie setzt auf eine systematische und konzeptionelle Trennung zwischen Umgebungsmodell und Störungen auf.

Um die Testansteuerungen und -auswertungen zu spezifizieren, bieten sich vielfältige Techniken und Testsprachen an. Jedoch sind sie meist proprietär und werkzeugspezifisch und können deshalb nur schwer phasenübergreifend eingesetzt oder gar für eine Zulieferer-Hersteller übergreifende Dokumentation verwendet werden. In der Telekomunikation hat sich deshalb ein Standard etabliert, TTCN-3 [1-3], der diese Lücke füllt. Um ihn auch auf regelungstechnische Systeme anwenden zu können, wurde er zu TTCN-3 Embedded erweitert [4].

Eine allgemeine Architektur für den Closed-Loop-Test

Zunächst einmal legen die Erfahrungen aus der modellbasierten Entwicklung nahe, zusätzliche Schichten für die Eingaben und Ausgaben

bereitzustellen, um so eine lose Kopplung des Umgebungsmodells zum Test zu erzielen. Einfache Umbenennungen oder auch Berechnungen können so ohne Modifikation des Umgebungsmodells adaptiert werden.

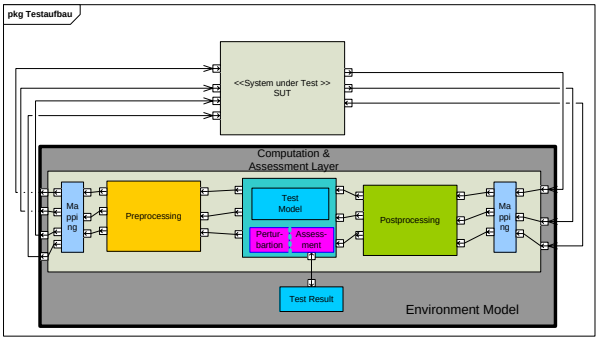

Abbildung 2: Generischer Testaufbau

Eine erfolgreiche Verwaltung von Testdaten setzt eine geeignete Beschreibung voraus, die insbesondere den Test eindeutig durch endlich viele handhabbare Daten erfasst. Für den Closed-Loop-Test spalten wir dazu das eigentliche Umgebungsmodell auf in ein generisches Verhalten und gezielte Störungen, die wir für den Test injizieren mit ihren Ergebniserwartungen. Betrachten wir dazu die Drosselklappenansteuerung. Rückgekoppelt mit dem Motormodell ist dieses System geschlossen und vollständig bestimmt. Nach einem initialen Anstoß liefe das geschlossene System in der Simulation eigenständig endlos weiter. Um nun dezidiert Tests durchzuführen, müssen wir in das selbstlaufende System eingreifen. Dazu definieren wir geeignete Schnittstellen für das Motormodell um zum Einen bestimmte Daten für eine Auswertung abgreifen zu können, zum Anderen, um weitere Testeingaben zu machen, die den eigenständigen Lauf des Systems stören.

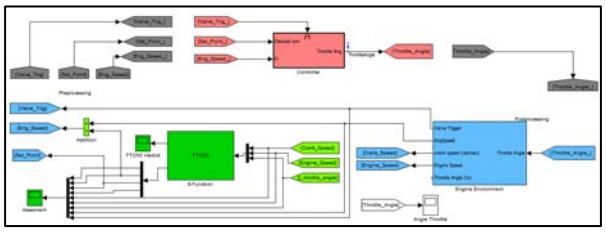

Abbildung 3: Closed-Loop-Test mit MATLAB

In Abbildung 3 ist das Testobjekt, der neue Drosselklappen Controller, rot markiert, das Umgebungsmodell, der Motor, blau. Über geeignete Schnittstellen, in diesem Fall Ausgänge des Motormodells, können der Verlauf verfolgt und beurteilt und gegebenenfalls auch die folgenden Störeingaben beeinflusst werden. Um weitere Eingaben zu machen, wurde der Ausgabenfluss des Motormodells unterbrochen, so dass über einen arithmetischen Block auf die Standardausgaben Störungen gesetzt werden können. Addiert man beispielsweise auf die Motorgeschwindigkeit ein geeignetes Rauschen, so lässt sich überprüfen, wie der Controller auf ein Rauschen der Sensorik reagiert. Diese Blöcke und Störeingaben sind hier grün markiert.

Für die Definition der Testeingaben und -erwartungen setzen wir TTCN-3 Embedded ein. Damit lassen sich die Aufwände für die Testspezifikation in verschiedenen Kontexten wieder nutzen.

TTCN-3 Embedded, Grundlegende Konzepte

Im Rahmen des Forschungsprojekts TEMEA wird eine Spracherweiterungen für die standardisierte Testspezifikationssprache TTCN-3 vorgeschlagen, die speziell die auf das Testen von Systemen mit kontinuierlichem und hybridem Echtzeitverhalten zugeschnitten sind.

TTCN-3 ist eine prozedurale Testsprache, Testverhalten wird in ihr algorithmisch definiert über das Senden und Empfangen von Daten über ports. Um auch auf regelungstechnische Systeme anwendbar zu sein, wurden die ursprünglichen Konzepte um Notationen wie **time**, **sampling** für die Echtzeitbeschreibung erweitert, aber auch um **stream**, **stream port** und **stream variable** für kontinuierliche Datenströme. Ohne zu sehr ins Detail zu gehen, soll hier beispielhaft gezeigt werden, wie kontinuierliche Testdateneingaben definiert werden können, beziehungsweise abgegriffen:

```
type port FloatOutPortType stream { out float }
    with {stepsize '0.001'};// alle 0.001 [s] sendet
dieser Port einen Wert

type port FloatInPortType stream { in float }
    with {stepsize '0.001'};

component tester{

    port FloatInStream engine_speed;

    port FloatOutStream engine_perturbation;

}

testcase myTestcase runs on tester{
  cont{
    engine_perturb:= 2000.0;
     assert(engine_speed < 4000.0);
  } until (duration > 10.0);
}
```

Das oben dargestellte Testprogramm definiert eine Testkomponente mit einem Eingangskanal und einem Ausgangskanal. Der Testfall stimuliert das System über den Ausgangskanal (***engine_perturb***) für die Dauer von 10.0 Sekunden und prüft über die gesamte Zeit, ob die Motordrehzahl den Wert 4000.0 Umdrehungen pro Minute nicht übersteigt.

Zur Zeit werden diese Erweiterungen für eine Übernahme in den TTCN-3 Standard geprüft.

Testmanagement und Wiederverwendbarkeit

Ein Testfall wird eindeutig definiert durch die Angabe des Umgebungsmodells und die Störeingaben mit Erwartungen. Es bietet sich an, die verschiedenen Umgebungsmodelle, die verschiedenen allgemeinen oder generischen Testszenarien entsprechen, in einer Bibliothek gesondert zu halten [5]. Zur geeigneten Sicherung dient ein Repository mit Versionsverwaltung.

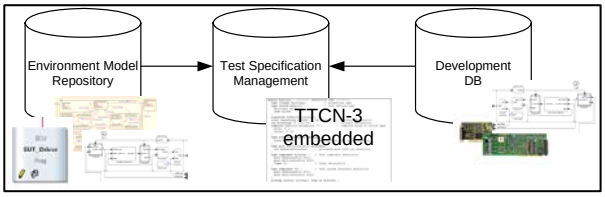

Abbildung 4: Testverwaltung

Durch ihre eindeutige Semantik und Standardisierung sind TTCN-3 Spezifikationen werkzeug- und herstellerübergreifend einsetzbar und sollten deshalb gesondert verwaltet werden. Der Aufwand für eine Anbindung von TTCN-3 ist einmalig für jedes Wekzeug. Für eine Reihe von Werkzeugen existieren bereits solche Anbindungen. Eine Erweiterung auf TTCN-3 Embedded ist allerdings noch zu implementieren und soll im Rahmen von TEMEA [4] für die Werkzeuge MATLAB/Simulink und CANoe exemplarisch durchgeführt werden.

Im Folgenden soll nun skizziert werden, wie dieser Ansatz effektiv zu einer Wiederverwendbarkeit von Testspezifikationen auch im Closed-Loop Fall eingesetzt werden kann.

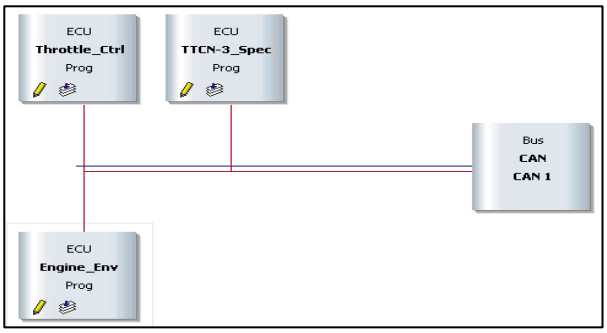

Abbildung 5: Closed-Loop Test mit CANoe

Weit verbreitet in der modellbasierten Entwicklung von Steuergeräten im Automobilbereich ist MATLAB/Simulink/StateFlow, [6]. Doch wird mit diesem Werkzeug nicht nur die zukünftige Software entwickelt, sondern meist auch für das Testen ein geeignetes Umgebungsmodell für den geplanten Einsatz des neuen Systems, siehe Abbildung 5.

Im weiteren Entwicklungsprozess wird aus dem getesteten Modell C-Code generiert und auf ein Steuergerät geflasht. Dieses Steuergerät kommuniziert über Bussysteme mit seiner Umgebung, in unserem Fall beispielsweise über CAN. Ein sehr verbreitetes Werkzeug für die Steuergeräteentwicklung in Netzwerken ist CANoe [7]. In ihm kann das neue Steuergerät über eine Buskommunikation mit anderen Geräten verbunden werden. Dabei können die weiteren Kommunikationsteilnehmer virtuell sein, z.B., weil die beteiligten Steuergeräte noch nicht verfügbar sind, oder aber auch reale Kontroller.

Für unser Testszenario können wir nun das generische Testszenario, das ausgewählte Umgebungsmodell, kompilieren und das CANoe-MATLAB-Interface nutzen, um unser Modell als virtuelles Steuergerät für den Test in CANoe einzubinden.

Die Stör- und Überwachungskomponenten für den Test binden wir ebenfalls als virtuellen Simulationsknoten ein. Durch die Abstraktion von den technischen Schnittstellen und die konzeptionelle Trennung in ein generisches Szenario und Störungen desselben für den Test erhalten wir so ein hohes Maß von Wiederverwendbarkeit. Umgebungsmodell und Test können über Adapter an die jeweils spezifischen technischen Schnittstellen angepasst werden und so direkt unter CANoe ablaufen.

Zusammenfassung und Ausblick

Closed-Loop-Tests sind für die analytische Absicherung regelungstechnischer Systeme unerläßlich. Um sie effektiv zu verwalten, wurde ein generischer Testaufbau vorgeschlagen, bei der das Umgebungsmodell in ein generisches Verhalten und in einen Störanteil aufgetrennt wird. Testeingaben entsprechen injizierten Störungen und werden TTCN-3 konform spezifiziert, ebenso die Erwartungen. Auf diese Weise lassen sich die verschiedenen Artefakte zur Testspezifikation verwalten und wiederverwenden.

Danksagungen

Wir möchten uns bei J. Grabowski, S. Sadeghipour, I. Schieferdecker, F.-W.. Schroer und J. Svacina für die zahlreichen Diskussionen und Anregungen bedanken, die in diese Arbeit eingeflossen sind. Diese Arbeit wurde im Rahmen des Forschungsverbunds TEMEA durch die Investitionsbank Berlin (IBB) gefördert. Dieses Programm wird von der EU kofinanziert. Die Mittel stammen aus dem Europäischen Fonds für Regionale Entwicklung/ EFRE,

Referenzen

[1] ETSI: ES 201 873-4 V.3.2.1: Methods dor Testing and Specification (MTS), The Testing and Test Control Notation Version 3, Part 4: TTCN-3 Operational Semantics. Sophia Antipolis, France, Febr. 2007

[2] ETSI: ES 201 873-4 V.3.2.1: Methods dor Testing and Specification (MTS), The Testing and Test

Control Notation Version 3, Part 5: TTCN-3 Runtime Interfaces. Sophia Antipolis, France, Febr. 2007

[3] C.Wilcox, Th. Deiß, et.al.: An Introduction to TTCN-3, John Wiley & Sons, 2005

[4] TEMEA (Testspezifikationstechnologie und -methodik für eingebettete Echtzeitsysteme im Automobil, *www.temea.org*

[5] SiLEST (Software in the Loop for Embedded System Test), Forschungsprojekt, *www.silest.de*

[6] TheMathWorks: Web Pages of Simulink-Simulation and Model-Based Design, *www.mathworks.com/products/simulink*

[7] Vector Informatik: Web Pages for CANoe *www.vector.com/vi_canoe_de*