# Objektorientierte Modellierung mit Modelica zur Echtzeitsimulation und Optimierung von Antriebssträngen

## Object-oriented Modelling with Modelica for Realtime Simulation and Optimization of Powertrains

**C. Schweiger**, **M. Otter**, Oberpfaffenhofen; **G. Cimander**, München

**Kurzfassung:**

*In diesem Beitrag werden Methoden und Werkzeuge zur Automatischen Applikation elektronischer Getriebesteuergeräte und deren Integration beschrieben. Am Beispiel eines mit der Sprache Modelica modellierten und in Echtzeit simulierten Sechsgang-Automatikgetriebes wird gezeigt, wie die Parameter eines exemplarischen Getriebesteuergerätes mittels numerischer Optimierung verbessert werden können.*

**Abstract:**

*In this paper, methods and tools for automatic application of electronic transmission control units are described in conjunction with their integration. At hand of a six-speed automatic gearbox, which is modelled with the language Modelica and simulated in realtime, it is shown, how the parameters of an exemplary transmission control unit can be improved by numerical optimization.*

### 1. Einleitung

Die sich ständig verkürzenden Entwicklungszyklen in der Automobilindustrie erfordern an diese Situation angepaßte Entwicklungsmethoden. Eine wichtige Rolle spielt dabei die Automatische Applikation elektronischer Steuergeräte (ESG). Sie sind im allgemeinen proprietäre Komponenten, so daß die Verwendung zahlreicher leistungsfähiger Reglerentwurfsmethoden nicht angewendet werden kann. Daher werden derzeit die Steuergeräteparameter häufig noch manuell im Fahrversuch oder an einem Prüfstand gesetzt. Bei Automatischer Applikation dagegen wird das Steuergerät in eine virtuelle Umgebung eingebettet, wie dies auch bei Hardware-in-the-Loop-Simulation geschieht. Die Steuergeräteparameter können dann mittels Optimierungsalgorithmen festgelegt werden.

Wesentliche Voraussetzung für Automatische Applikation ist folglich das Vorliegen echtzeitfähiger Modelle. Insbesondere zur Modellierung der in Antriebssträngen enthaltenen Reibelemente wie Kupplungen, Bremsen oder Zahneingriffe hat sich hierfür die hybrid diskret-

kontinuierliche Modellierungstechnik etabliert, die unter anderem von der Modellierungssprache MODELICA® unterstützt wird [1]. In Kombination mit leistungsfähigen Simulationswerkzeugen [2] ist unter Ausnutzung symbolischer Gleichungsvorverarbeitung, automatischer Rechencode-Generierung und intelligenter Integratoren die Echtzeitsimulation selbst komplexer Modelle möglich [3]. Zudem erlaubt der objektorientierte Ansatz von Modelica eine sehr anschauliche Vorgehensweise: Komponenten werden aus Bibliotheken entnommen und an ihren physikalischen Schnittstellen zu einem hierarchisch aufgebauten Gesamtsystem verbunden.

Erleichtert wird die Modellierung von Antriebssträngen durch die vom DLR entwickelte Bibliothek PowerTrain [4]. Unter anderem stehen neben zahlreichen Einzelkomponenten anpaßbare Modelle von Gesamtsystemen zur Verfügung. Hervorzuheben ist die effiziente und robuste Behandlung von geschwindigkeits- und momentenabhängiger Reibung [5], wie sie beispielsweise zur Modellierung des Verzahnungswirkungsgrades von Getrieben benötigt wird.

Im weiteren stehen kommerziell erhältliche Software-Werkzeuge zur Verfügung, die es erlauben, aus den Antriebsstrang-Modellen C-Code zu erzeugen, der im Anschluß auf einem Echtzeitrechensystem ausgeführt wird. Betrachtet wird die Applikation eines Automatikgetriebe-Steuergeräts, das verschiedene vom Modell berechnete Fahrzustände auswertet und daraus Kupplungsdrücke ermittelt, die wiederum dem Modell zur Verfügung gestellt werden.

Zur automatischen Bestimmung der Steuergeräteparameter wurde die vom DLR entwickelte Optimierungsumgebung Multi-Objective Parameter Synthesis (MOPS) [6] an das Echtzeitrechensystem angebunden [7]. Dabei werden im Modell verschiedene, die Zielsetzungen Agilität, Komfort und Kraftstoffersparnis repräsentierende Kriterien berechnet und vom jeweiligen Optimierungsalgorithmus iterativ zur Berechnung verbesserter Steuergeräteparameter herangezogen.

Im Unterschied zur beschriebenen, angestrebten Vorgehensweise ist kein Hardware-Steuergerät verwendet worden. In das Gesamtmodell ist lediglich ein virtuelles Steuergerät integriert. Dieses ist jedoch durch Schnittstellen derart abgegrenzt, daß ein Austausch gegen Hardware problemlos möglich ist. Nachdem die Anbindung von Steuergeräten an Echtzeitrechensysteme kommerziell bezogen werden kann, wurde diese vorerst nicht betrachtet.

## 2. Modellierung mit Modelica

Modelica ist eine frei verfügbare, objektorientierte Sprache zur Modellierung großer, komplexer und heterogener physikalischer Systeme. Sie ist zur domänenübergreifenden Modellierung geeignet, beispielsweise zur Erstellung mechatronischer Modelle für Robotik-, Automo-

bil- und Luftfahrtanwendungen unter Einbeziehung mechanischer, elektrischer, hydraulischer, thermodynamischer und regelungstechnischer Subsysteme.
Modelle in Modelica sind mathematisch mit Differentialgleichungen, algebraischen und diskreten Gleichungen beschrieben, ohne daß manuell nach einer bestimmten Variable aufgelöst werden muß. Einem Modelica-Werkzeug werden genügend Informationen bereitgestellt, um dies automatisch zu entscheiden. Mit Verwendung spezieller Algorithmen zur effizienten Behandlung großer Modelle können mehr als hunderttausend Gleichungen bewältigt werden.
In Modelica werden sowohl detaillierte, gleichungsorientierte Modellierung zur Bibliothekskomponentenerstellung als auch die Wiederverwendung letzterer zur Komposition von hierarchisch aufgebauten Modellen unterstützt. Modelle von Standardkomponenten sind typischerweise in Modellbibliotheken verfügbar. Mit einem graphischen Modelleditor kann ein Modell erstellt werden, indem Modellbausteine nebeneinander angeordnet, Verbindungen gezeichnet und Parameter in Dialogfelder eingegeben werden. Bild 1 zeigt ein typisches Modelica-Modell.

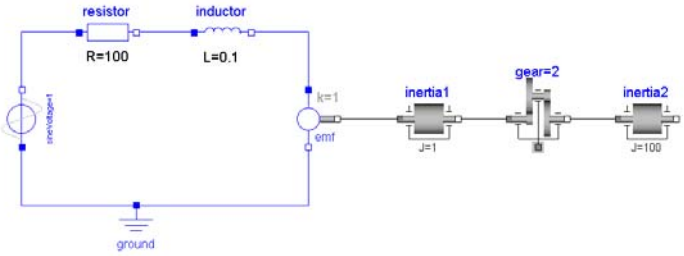


Bild 1: Typisches Modelica-Modell

Hervorzuheben ist, daß die Verbindungen zwischen den Komponenten keine Signalflüsse, sondern physikalische Verbindungen von Konnektoren darstellen. Ein Konnektor enthält alle notwendigen physikalischen Größen, um die Interaktion mit anderen Komponenten zu beschreiben. Beispiele von Konnektoren sind in Tabelle 1 wiedergegeben.
Werden mehrere Konnektoren miteinander verbunden, so resultieren daraus Gleichungen, welche besagen, daß die Potentialvariablen in den beteiligten Konnektoren gleich sind und sich die Flußvariablen der beteiligten Konnektoren zu Null aufsummieren. Weitere Gleichungen sind in den Komponenten selbst enthalten. Tabelle 2 zeigt den Modelica-Text für eine rotatorische Massenträgheit, wie sie im Modell in Bild 1 zweifach enthalten ist.
Basierend auf Typen gemäß ISO 31-1992 werden ein Parameter für das Massenträgheitsmoment und Variablen für Winkel, Winkelgeschwindigkeit und Winkelbeschleunigung deklariert. Die beiden *Flansche* genannten Konnektoren dienen zum Verbinden zu anderen Komponenten. Im Gleichungsteil wird im wesentlichen die Differentialgleichung 2. Ordnung für

Tabelle 1: Verschiedene Konnektordefinitionen

| | Potentialvariable | Flußvariable | Aussehen |
|---|---|---|---|
| Elektrik | Elektrisches Potential | Strom | PositivePin<br>NegativePin |
| Mechanik, translatorisch | Position | Kraft (skalar) | Flange_a<br>Flange_b |
| Mechanik, rotatorisch | Winkel | Moment (skalar) | Flange_a<br>Flange_b |
| Wärmeübertragung | Temperatur | Wärmestrom | HeatPort_a<br>HeatPort_b |
| Hydraulik (inkompressibel) | Druck | Volumenstrom | Port_A<br>Port_B |
| Pneumatik (kompressibel) | Druck | Massenstrom | Port_1<br>Port_2 |
| Mehrkörperdynamik | Positionsvektor, Transformationsmatrix | Schnittkraft (vektoriell), Schnittmoment (vektoriell) | Frame_a<br>Frame_b |
| Mehrphasen-Wärmeströmung | Druck, spezifische Enthalpie, Massenanteil | Gesamt-/Einzelmassen-ströme, Enthalpiestrom, | FluidPort_a<br>FluidPort_b |
| Signalflüsse | Signalvektor (Real, Integer, Boolean) | — | InPort<br>OutPort |

Tabelle 2: Modelica-Text für rotatorische Massenträgheit

```
model Inertia
  import SI = Modelica.SIunits;
  import Modelica.Mechanics.Rotational.Interfaces.Flange;

  parameter SI.Inertia J=1;

  SI.Angle phi;
  SI.AngularVelocity w;
  SI.AngularAcceleration a;

  Flange flange_a;
  Flange flange_b;
equation
  flange_a.phi = phi;
  flange_b.phi = phi;

  w = der(phi);
  a = der(w);
  J*a = flange_a.tau + flange_b.tau;
end Inertia;
```

eine Massenträgheit wiedergegeben. Der **der**-Operator drückt aus, daß die Winkelgeschwindigkeit die zeitliche Ableitung des Winkels und die Winkelbeschleunigung die zeitliche Ableitung der Winkelgeschwindigkeit ist. Die Momentenbilanz verdeutlicht, daß Gleichungen in Modelica nicht als Zuweisungen, sondern als Deklarationen zu verstehen sind: So finden sich auf der linken Seite der Momentenbilanz zwei Variablen, was beispielsweise in Programmiersprachen nicht zulässig wäre.

Für die Austauschbarkeit und Weitergabe von Modellen ist es wichtig, Bibliotheken verfügbar zu haben, die die am häufigsten verwendeten Komponenten enthalten. Aus diesem Grund wird von der Modelica Association die ständig wachsende Modelica-Standardbibliothek entwickelt und gepflegt, s. Bild 2. Neben weiteren freien Bibliotheken existieren auch kommerzielle Bibliotheken wie die im nächsten Abschnitt unter anderem vorgestellte PowerTrain-Bibliothek.

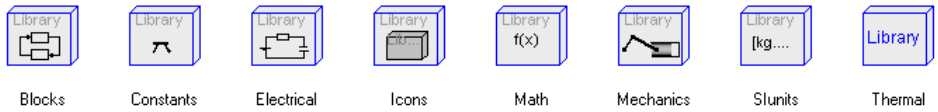


Bild 2: Bibliotheken der Modelica-Standardbibliothek

Um die Modellierungssprache und die Bibliotheken zu benutzen, wird Software benötigt, welche das Modelica-Modell in eine Form übersetzt, die in einer entsprechenden Simulationsumgebung effizient simuliert werden kann. Am weitesten verbreitet ist das Werkzeug Dymola, welches leistungsfähige Algorithmen zur symbolischen Gleichungsmanipulation beinhaltet. So wird der ursprünglich aus dem Modelica-Modell abgeleitete Satz von Gleichungen erheblich reduziert. Weiter sind Vorkehrungen zur Behandlung von Systemen mit höherem Index oder mit gemischten Gleichungen (boolsche und reelle Variablen) getroffen.

### 3. Modellierung von Antriebssträngen mit Modelica

Im folgenden wird beschrieben, wie Kraftfahrzeug-Antriebsstränge mit Modelica modelliert werden können. Der wesentliche, in Antriebssträngen zu modellierende Effekt ist Reibung. Sie ist zum Beispiel in einem Automatikgetriebe sowohl in den Kupplungen und Bremsen, als auch in der Verzahnung und den Lagern von Bedeutung.

Für die Zielsetzung der automatischen Applikation sind, wie eingangs erwähnt, echtzeitfähige Modelle erforderlich. Zur echtzeitfähigen Simulation von unter anderem Reibelementen hat sich der hybrid diskret-kontinuierliche Modellierungsansatz etabliert, welcher in Modelica besonderen Eingang gefunden hat. Modelica erlaubt die Behandlung von diskontinuierlichen und strukturvariablen Komponenten. Neben den bereits erwähnten fallen darunter Komponenten wie Relais, Schalter, Stoß, abgetastete Systeme etc. In Modelica wurden spezielle Sprachkonstrukte eingeführt, die einem Simulator erlauben, die dafür benötigte effiziente Ereignisbehandlung anzuwenden.

Zahlreiche auf die Verwendung in Antriebssträngen zugeschnittene Komponenten werden in der vom DLR entwickelten PowerTrain-Bibliothek bereitgestellt. Neben 45 Einzelkomponenten sind zehn zum Teil sehr detaillierte Beispielmodelle enthalten, die die Verwendung demonstrieren und dem Benutzer als Hilfestellung bei der Entwicklung eigener Gesamtmodelle

dienen. Eine Besonderheit der Bibliothek ist die effiziente und robuste Behandlung von geschwindigkeits- und lastabhängiger Reibung, welche unter detaillierter beschrieben wird.
Bei der Entwicklung der PowerTrain-Bibliothek wurde unter anderem Wert auf Benutzerfreundlichkeit gelegt. So wurde ein vom Benutzer anpaßbarer Signalbus eingeführt, der die ansonsten entstehenden komplexen Signalverbindungen zwischen den Einzelkomponenten vermeidet. Bild 3 zeigt ein Gesamtmodell eines Antriebsstranges, welches als zentrale Komponente den Signalbus beinhaltet. Die weiteren Komponenten weisen eine einzige Verbindung mit dem Signalbus auf. Sie ist ausreichend, um alle benötigten Signale auf den Bus zu senden oder von ihm zu empfangen.

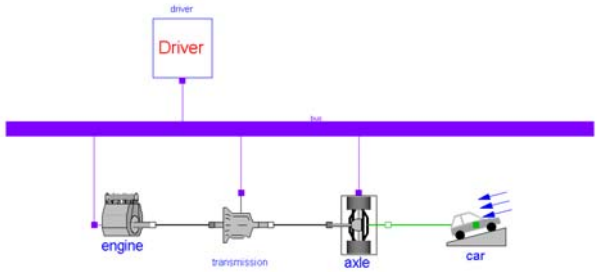


Bild 3: Modell DriveLine

Um ein zu starkes Anwachsen von Modellvarianten zu vermeiden, wurde es ermöglicht, bei den Antriebsstrang-Komponenten verschiedene Ausprägungen auszuwählen. Es gibt somit nur eine Getriebekomponente. Nach dem diese in das Modell eingefügt wurde, ermöglicht das in Bild 4 gezeigte Auswahlfenster die Angabe des tatsächlichen Getriebetyps.

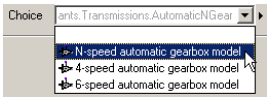


Bild 4: Auswahlfenster zur Festlegung des Getriebetyps

Für Plausibilitätsprüfungen und Demonstrationszwecke wurden alle Getriebekomponenten standardmäßig mit Animationsinformation und einer auf Antriebsstränge zugeschnittenen Parametrierung versehen. Die resultierende Ansicht ist in Bild 5 gezeigt. Um für Echtzeitanwendungen keine gesonderten Modelle anfertigen zu müssen, ist die Animation per Parameter abschaltbar mit der Folge, daß alle lediglich für Animation benötigten Gleichungen aus dem Modell entfernt werden.

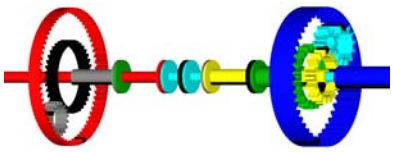

Bild 5: Animationsansicht eines Lepelletier-Radsatzes

Im folgenden wird die eingangs erwähnte robuste und effiziente Behandlung geschwindigkeits- und momentenabhängiger Reibung am Beispiel eines einfachen Getriebes beschrieben. In [5] wird dieses Reibungsmodell als

$$\tau_B = i(-\tau_A + \Delta\tau) \qquad \text{mit} \qquad \Delta\tau = \begin{cases} (1-\hat{\eta}_{mf})\tau_A + \hat{\tau}_{bf} & : \ \omega_A \neq 0 \\ \text{so,daß } \dot{\omega}_A = 0 & : \ \omega_A = 0 \end{cases}$$

hergeleitet, wobei $\tau_A$ und $\tau_B$ Flanschmomente, $i$ die Getriebeübersetzung und $\Delta\tau$ das Reibmoment darstellen. Die Winkelgeschwindigkeit von Flansch A ist $\omega_A$. Das Reibmoment wird für den Gleitzustand ($\omega_A \neq 0$) und den Haftzustand ($\omega_A = 0$) unterschiedlich berechnet. Während es im Gleitzustand in Abhängigkeit von Verzahnungswirkungsgrad $\hat{\eta}_{mf}$ und additiver Lagerreibung $\hat{\tau}_{bf}$ berechnet wird, wird es im Haftzustand implizit so bestimmt, daß die Winkelbeschleunigung Null ist und der Haftzustand erhalten bleibt.

Eine Umschaltung zwischen beiden Zuständen erfolgt folgendermaßen: Vom Haft- in den Gleitzustand wird umgeschaltet, wenn die Differenz der äußeren Momente das aus $\hat{\eta}_{mf}$ und $\hat{\tau}_{bf}$ bestimmte Haftreibungsmoment überschreitet. Umgekehrt wird vom Gleit- in den Haftzustand umgeschaltet, wenn $\omega_A = 0$ erfüllt ist.

Die Größen $\hat{\eta}_{mf}$ und $\hat{\tau}_{bf}$ werden in Abhängigkeit von der Leistungsflußrichtung ($\operatorname{sgn}\tau_A\omega_A = \pm 1$) und der Winkelgeschwindigkeit $\omega_A$ bestimmt, was zusammengefaßt in Tabelle 3 wiedergegeben ist. Dabei stellen die Indizes $i = 1,2$ bei $\eta_{mfi}$ und $\tau_{bfi}$ die Zuordnung zur entsprechenden Leistungsflußrichtung her.

Tabelle 3: Berechnung des Reibmomentes im Gleitzustand

| | $\tau_A \geq 0$ | $\tau_A < 0$ |
|---|---|---|
| $\omega_A > 0$ | $\Delta\tau = \left[1-\eta_{mf1}(\omega_A)\right]\tau_A + \lvert\tau_{bf1}\rvert$ | $\Delta\tau = \left[1-1/\eta_{mf2}(\omega_A)\right]\tau_A + \lvert\tau_{bf2}\rvert$ |
| $\omega_A < 0$ | $\Delta\tau = \left[1-1/\eta_{mf2}(\omega_A)\right]\tau_A - \lvert\tau_{bf2}\rvert$ | $\Delta\tau = \left[1-\eta_{mf1}(\omega_A)\right]\tau_A - \lvert\tau_{bf1}\rvert$ |

Graphisch läßt sich die Abhängigkeit des Reibmoments vom Lastmoment und der Winkelgeschwindigkeit wie in Bild 6 gezeigt darstellen. Die verschiedenen Bereiche zur Berechnung von $\Delta\tau$ sind im linken Teilbild ersichtlich. Im Gleitzustand nimmt $\Delta\tau$ abhängig vom Vorzeichen von $\omega_A$ entweder einen Wert auf der oberen oder unteren Begrenzungslinie an. Mit der Maßgabe $\dot{\omega}_A = 0$ stellt sich im Haftzustand ein Wert zwischen den beiden Begrenzungsli-

nien ein. Das mittlere Teilbild zeigt $\Delta\tau$ mit $\omega_A$ als Abszisse und $\tau_A$ als Lageparameter. Deutlich ist die Verwandschaft zu einem rein geschwindigkeitsabhängigen Reibmodell erkennbar. Der einzige Unterschied besteht in der zusätzlichen Abhängigkeit des Reibmomentes vom Lastmoment $\tau_A$. Eine kombinierte Darstellung zeigt das rechte Teilbild.

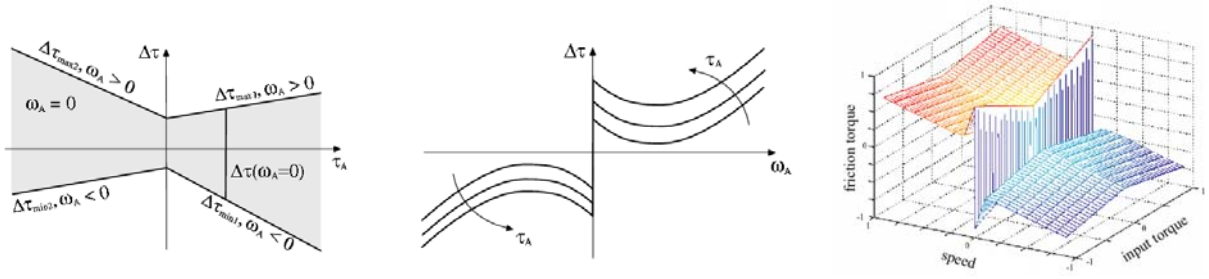


Bild 6: Reibmoment im Gleit- und Haftzustand

Dieses Reibmodell wurde in Modelica implementiert und wird in vielen Komponenten der PowerTrain-Bibliothek verwendet. Die Abhängigkeit der Größen $\eta_{mf1,2}$ und $\tau_{bf1,2}$ von $\omega_A$ wird mittels Interpolation in einer vom Benutzer bestimmbaren Tabelle realisiert. Die Robustheit des Reibmodells ergibt sich aus zwei Umständen. Zunächst werden durch die besondere Formulierung nichtlineare Gleichungssysteme vermieden, für welche nur unzureichende Löser existieren würden. Weiter wird der Haftzustand explizit berücksichtigt, was bei vergleichbaren Getriebewirkungsgradmodellen nicht der Fall ist. Bei Verwendung von Lösern mit variabler Schrittweite zur Simulation solcher Modelle ist dann häufig eine Reduzierung der Schrittweite gegen Null beobachtbar, was in der Regel zum Abbruch führt.

## 4. Echtzeitsimulation

Die Software Dymola erlaubt es, Modelica-Modelle in einen Satz mathematischer Gleichungen überzuführen, welcher symbolisch vorverarbeitet als optimierter C-Code ausgegeben wird. Im speziellen wird auch die Einbettung dieses C-Codes als S-Function in Simulink [8] unterstützt. Eine mögliche Vorgehensweise besteht also darin, zur Modellierung Modelica mit Dymola einzusetzen und das erhaltene Modell im weiteren Entwurfs- und Analyseprozeß als einen E/A-Block in Simulink zu verwenden.

Diese Vorgehensweise bot sich somit auch für die Echtzeitsimulation an. Als Echtzeitrechensystem wurde dazu xPC Target [8] verwendet, welches auf einem Standard-PC von Diskette gebootet werden kann. Als Bindeglied zwischen Simulink und dem Echtzeitrechensystem fungiert Real-Time Workshop [8], womit aus dem Simulink-Modell direkt C-Code für eine gewünschte Zielplattform erstellt werden kann. xPC Target ermöglicht die Verwendung zahlreicher E/A-Hardware und stellt Treiber dafür zur Verfügung. Damit kann auch ein in

Hardware vorliegendes Steuergerät angeschlossen werden, womit der Übergang zur Hardware-in-the-Loop-Simulation vollzogen wäre.

## 5. Optimierung von Steuergeräteparametern mit MOPS

Der noch fehlende Schritt von der Hardware-in-the-Loop-Simulation zur Automatischen Applikation von Steuergeräten ist in der Automatisierung der Tätigkeit des Applikationsingenieurs zu sehen. Sie besteht darin, Steuergeräteparameter solange gezielt zu verändern, bis bestimmte Kriterien erfüllt sind. Wenn es möglich ist, die zu erfüllenden Kriterien zu quantifizieren, ist die entscheidende Voraussetzung erfüllt, um für diese Aufgabe numerische Optimierungsverfahren einzusetzen.

Am DLR wurde die Optimierungsumgebung Multi-Objective Parameter Synthesis (MOPS) [6] entwickelt, welche die Umsetzung von Entwurfsproblemen zu wohlformulierten, mehrzieligen Optimierungsaufgaben in MATLAB [8] unterstützt. Neben verschiedenen leistungsfähigen Optimierungsalgorithmen sind vordefinierte regelungstechnischen Kriterien ebenso wie generische Skripte zur Ablaufsteuerung enthalten. Zudem stehen Visualisierungswerkzeuge zur Überwachung des Entwurfsprozesses zur Verfügung. Weitere Funktionen helfen bei der Bewältigung großer Parameter- und Kriterienmengen, ermöglichen bei zeitaufwändigen Berechnungen verteiltes Rechnen und erlauben die Verwendung externer Simulations- und Analyseserver. Parameterschätzung für Identifikationsprobleme wird ebenso unterstützt wie optimierungsbasierte Robustheitsanalyse.

In [7] ist die Anbindung der Software MOPS an das Echtzeitrechensystem beschrieben. Als Vorteil erwies sich dabei die Durchgängigkeit der Werkzeuge. Alle beim in Bild 7 gezeigten Aufbau beteiligten Komponenten können von MATLAB bzw. Simulink aus angesprochen werden.

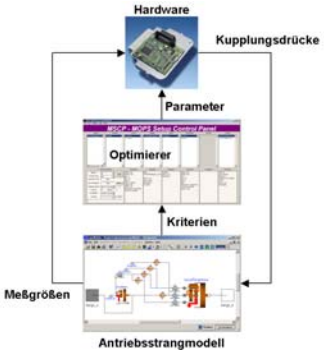


Bild 7: Aufbau einer Entwurfsumgebung für Automatische Applikation

## 6. Anwendungsbeispiel

Zur Veranschaulichung werden ein typisches, mit der PowerTrain-Bibliothek erstelltes Antriebsstrangmodell vorgestellt und exemplarische Ergebnisse aus [7] gezeigt.

Ausgehend von der in Bild 3 gezeigten Komponente *transmission* zeigt das linke Teilbild von Bild 8 deren Inhalt auf der darunterliegenden Hierarchieebene. Neben dem eigentlichen Getriebe sind als Submodelle ein hydrodynamischer Drehmomentwandler, Steuergeräte für Gangwahl und Ansteuerung von Kupplungen und Bremsen des Getriebes und der Wandlerüberbrückungskupplung erkennbar. Die nächsttiefere Hierarchieebene zeigt das rechte Teilbild von Bild 8, in dem der Inhalt der Komponente *lepelletier* dargestellt ist.

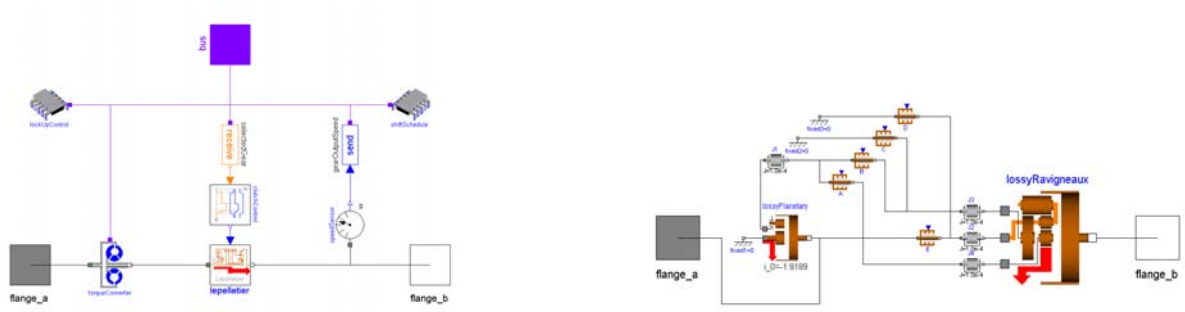


Bild 8: Modellstruktur für Sechsgang-Automatikgetriebe

Der Lepelletier-Radsatz [9] besteht aus einem Planetengetriebe und einem Ravigneaux-Radsatz. Mit dem Präfix *lossy* wird angedeutet, daß das in Abschnitt 3 skizzierte Reibmodell beinhaltet ist. Hervorzuheben ist, daß auch in den Kupplungen A, B, E und den Bremsen C, D detaillierte Reibmodelle verwendet werden. Die Simulation derart verschalteter Reibelemente ist kein triviales Problem, erweist sich bei Verwendung von Modelica/Dymola jedoch als besonders effizient und robust.

In [7] sind Ergebnisse gezeigt, die unter Verwendung eines ähnlichen Modells das Zusammenwirken der verschiedenen Einzelwerkzeuge demonstrieren. Auszüge davon sind in Bild 9 gezeigt.

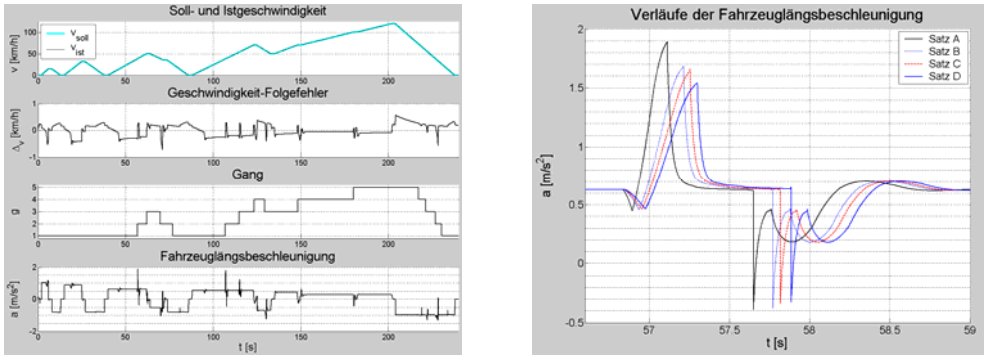


Bild 9: Ergebnisse von Simulation und Optimierung

Das linke Teilbild zeigt Zeitverläufe von Soll- und Istgeschwindigkeit, deren Differenz, den gewählten Gang und die Fahrzeuglängsbeschleunigung für *einen* Parametersatz. Im rechten Teilbild ist gezeigt, wie sich die Fahrzeuglängsbeschleunigung bei einem Schaltvorgang für verschiedene vom Optimierer erzeugte Parametersätze verändert.

## 7. Zusammenfassung

Das Zusammenwirken von objektorientierter Modellierung mit Modelica, automatischer Code-Generierung, Echtzeitsimulation und numerischer Optimierung mit MOPS ließ eine leistungsfähige Umgebung zur Automatischen Applikation elektronischer Steuergeräte entstehen. Trotz der Verwendung mehrerer unabhängiger, auf die jeweilige Teilaufgabe spezialisierte Werkzeuge ist die für Produktionsumgebungen nötige Durchgängigkeit gegeben.

## Danksagung

Die Autoren danken Prof. Dr. B. Heißing und Dr. J. Fink vom Lehrstuhl für Fahrzeugtechnik der Technischen Universität München für die Übernahme der Betreuung und die fachliche Unterstützung bei der Anfertigung von [7].

Im Rahmen des Projekts *Test und Optimierung elektronischer Fahrzeug-Steuergeräte mit Hardware-in-the-Loop-Simulation*, AZ300-3245.2-3/01, wurde die hier beschriebene Arbeit teilweise vom *Bayerischen Staatsministerium für Wirtschaft, Infrastruktur, Verkehr und Technologie* unterstützt.

## Literaturverzeichnis

[1] Otter, M., Schlegel, C.: Objektorientierte Modellierung Physikalischer Systeme, Teil 9, Modellierung von Antriebssträngen. at Automatisierungstechnik 47 (1999) 12 S. A33-A36

[2] Dymola. URL http://www.dynasim.se/

[3] Elmqvist, H., Mattsson, S. E., Olsson, H., Andreasson, J., Otter, M., Schweiger, C., Brück, D.: Real-time Simulation of Detailed Automotive Models. MODELICA ‘03, S. 29-38

[4] Otter, M., Dempsey, M., Schlegel, C.: Package PowerTrain. A Modelica Library for Modeling and Simulation of Vehicle Power Trains. MODELICA ’00, S. 23-32

[5] Pelchen, C., Schweiger, C., Otter, M.: Modeling and Simulating the Efficiency of Gearboxes and of Planetary Gearboxes. MODELICA ’02, S. 257-266

[6] Joos, H.-D., Bals, J., Looye, G., Schnepper, K., Varga, A.: A multi-objective optimisation-based software environment for control systems design. CACSD ’02, S. 7-14

[7] Cimander, G.: Echtzeitsimulation und Optimierung eines Automatikgetriebes. Lehrstuhl für Fahrzeugtechnik, Technische Universität München, Diplomarbeit. München 2003

[8] The MathWorks. URL http://www.mathworks.com/

[9] Dach, H., Gruhle, W.-D., Köpf, P.: Pkw-Automatgetriebe. Landsberg/Lech: Verlag Moderne Industrie 2001