14. SYMPOSIUM „DESIGN FOR X“
NEUKIRCHEN, 13. UND 14. OKTOBER 2003

WIRKPRINZIPIEN DER SELBSTOPTIMIERUNG

Andeas Schmidt, Jürgen Gausemeier

Kurzfassung

Die meisten Erzeugnisse des Maschinenbaus beruhen schon heute auf dem engen Zusammenwirken von Mechanik, Elektronik, Regelungstechnik und Softwaretechnik, was durch den Begriff Mechatronik zum Ausdruck kommt. Ziel der Mechatronik ist es, das Verhalten eines technischen Systems zu verbessern. Mit Hilfe von Sensoren werden Informationen über die Umgebung, aber auch über das System selbst erfasst und für eine optimale Reaktion des Systems verarbeitet. Vor dem Hintergrund der rasanten Entwicklung der Informationstechnik zeichnen sich weitere Möglichkeiten ab, die weit über die Mechatronik hinausgehen – sich selbst optimierende Systeme mit inhärenter Teilintelligenz.

Der vorliegende Beitrag stellt Wirkprinzipien der Selbstoptimierung als Verhaltensmuster intelligenter Systemelemente vor. Der Rückgriff auf Wissensbasen von Wirkprinzipien erlaubt den Systemelementen, Erfahrungen, die in der Vergangenheit gemacht wurden auf die aktuelle Situation anzupassen, entsprechend einzusetzen und aus den Resultaten zu Lernen. Ein Wissensmodell sowie ein Vorgehensmodell zum erfahrungsbasierten Einsatz der Wirkprinzipien liefern eine methodische Basis des vorgestellten Ansatzes.

1 Einleitung

Aus der zunehmenden Durchdringung des Maschinenbaus mit Informationstechnik eröffnen sich erhebliche Erfolgspotentiale. Der Begriff Mechatronik bringt dies zum Ausdruck. Mechatronik beschreibt das enge Zusammenwirken von Mechanik, Elektronik, Regelungstechnik und Software. Ziel der Mechatronik ist es, das Verhalten eines technischen Systems zu verbessern, indem mit Hilfe von Sensoren Informationen über die Umgebung und das System selbst erfasst werden. Diese Informationen werden in Prozessoren verarbeitet, die im jeweiligen Kontext optimale Reaktionen mit Hilfe von Aktoren auslösen. Durch den Einbezug der modernen Informationstechnik in die Produkte selbst können anpassungsfähige technische Systeme entstehen. Diese Systeme sind in der Lage, auf Veränderungen in ihrer Umwelt zu reagieren und Abläufe, die nur schwer steuerbar sind, durch Regelungstechnik zu optimieren [1].

Künftige mechatronische Systeme werden aus Konfigurationen von Systemelementen mit inhärenter Teilintelligenz bestehen. Das Verhalten des Gesamtsystems wird durch die Kommunikation und Kooperation der intelligenten Systemelemente geprägt sein. Aus informationstechnischer Sicht handelt es sich nach unserem Verständnis um verteilte Systeme von miteinander kooperierenden Agenten. Daraus eröffnen sich faszinierende Möglichkeiten für die Gestaltung der maschinenbaulichen Erzeugnisse von morgen. Selbstoptimierung ermöglicht handlungsfähige Systeme mit inhärenter „Intelligenz“, die in der Lage sind, selbständig und flexibel auf veränderte Umgebungsbedingungen zu reagieren.

Für den Entwurf selbstoptimierender Systeme besteht der Handlungsbedarf, Wirkprinzipien im Kontext der Selbstoptimierung zu erforschen. Das Wirkprinzip in der traditionellen Konstruktionslehre bezeichnet den Zusammenhang vom physikalischen Effekt sowie geometrischen und stofflichen Merkmalen (Wirkgeometrie, Wirkbewegung und Wirkstoff). Es lässt das Prinzip der Lösung zur Erfüllung einer Teilfunktion erkennen [2]. Eine lösungsgetriebene

Modellbildung von Verhaltensweisen selbstoptimierender Systeme erfordert jedoch den Rückgriff auf verhaltensbestimmendes Erfahrungswissen in Form von Verhaltensmustern.

2 Selbstoptimierende Systeme des Maschinenbaus

Die Komplexität selbstoptimierender Systeme des Maschinenbaus wird durch die Zerlegung in eine hierarchische Funktionsstruktur bestehend aus Teilfunktionen $F_{i,j}$ beherrscht. Jeder Teilfunktion ist ein Wirkprinzip der Selbstoptimierung $WPso_{i,j}$ zugeordnet, das ein Verhaltensmuster zur Erfüllung der Teilfunktion repräsentiert. Lösungselemente mit inhärenter Teilintelligenz implementieren die Funktionalität der Wirkprinzipien. Das selbstoptimierende System besteht aus Konfigurationen dieser Lösungselemente. Das Verhalten des Gesamtsystems wird durch die Kommunikation und Kooperation der intelligenten Lösungselemente geprägt (Bild 1). Softwaretechnisch handelt es sich um verteilte Systeme von miteinander kooperierenden Agenten [3]:

> *Ein Agent ist ein autonomes, proaktives, kooperatives und hochgradig adaptives Funktionsmodul. Autonom impliziert eine eigenständige Kontrolle, die von sich aus Aktionen initiiert (proaktiv). Agenten werden als Funktionsmodule angesehen, die in Kooperation oder Konkurrenz zueinander handeln. Adaptiv bezeichnet ein zur Laufzeit generisches Verhalten, das beispielsweise auch Lernfähigkeit beinhalten kann. Ein Funktionsmodul wird als heterogenes Teilsystem mit elektronischen, mechanischen und informationstechnischen Komponenten verstanden.*

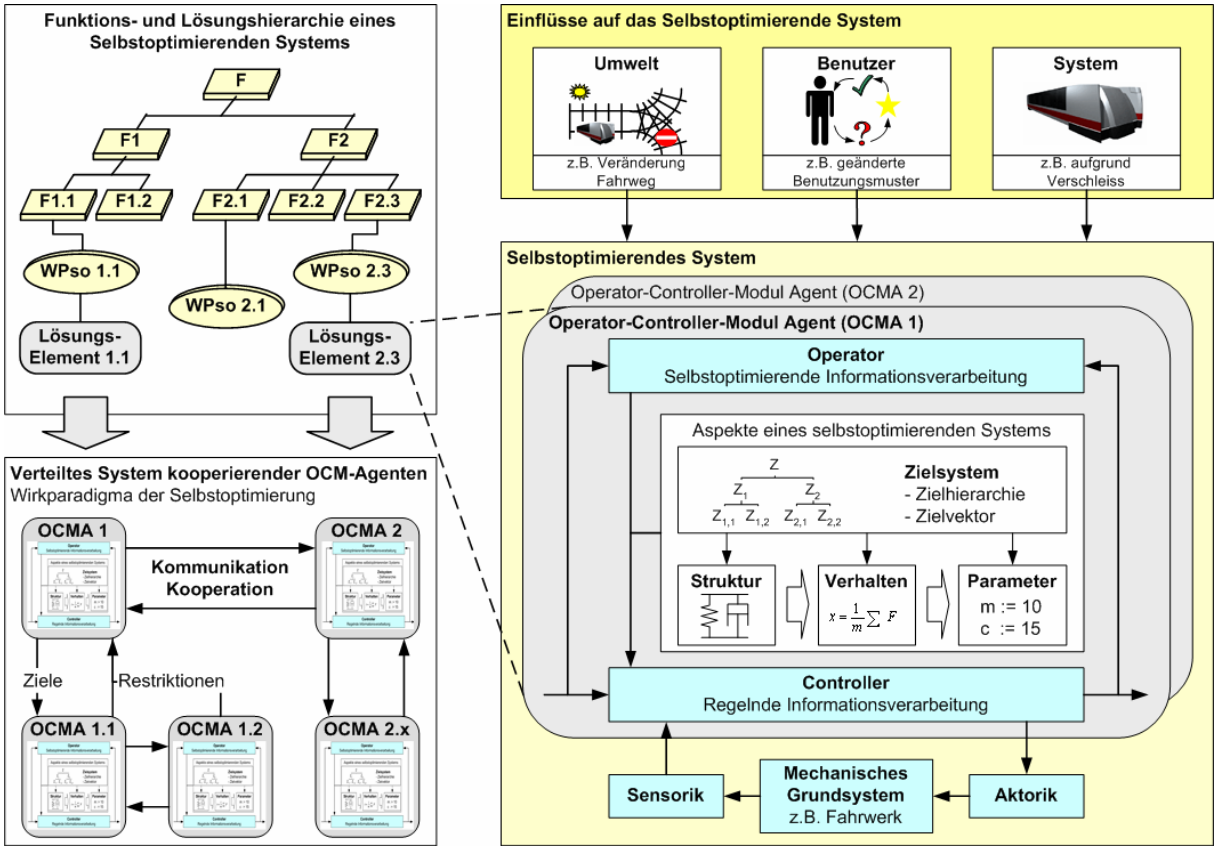

Bild 1: Der Weg vom Entwurf selbstoptimierender Systeme durch Zerlegung in eine Funktions- und Lösungshierarchie hin zum Einsatz kooperierender intelligenter Agenten

Die Verbindung des Paradigmas der intelligenten Agenten mit mechatronischen Strukturen ermöglicht selbstoptimierende maschinenbauliche Systeme.

> *Unter Selbstoptimierung eines technischen Systems wird die endogene Änderung des Zielvektors durch veränderte Umweltbedingungen und die daraus resultierende zielkonforme autonome Anpassung der Struktur, des Verhaltens sowie der Parameter dieses Systems verstanden. Damit geht Selbstoptimierung über die bekannten Regel- und Adaptionsstrategien wesentlich hinaus; Selbstoptimierung ermöglicht handlungsfähige Systeme mit inhärenter „Intelligenz", die in der Lage sind, selbständig und flexibel auf veränderte Umgebungsbedingungen zu reagieren. [3]*

Die Betrachtung von selbstoptimierenden Systemen beruht auf den vier Aspekten Zielsystem (z.B. hierarchisches System von Zielen bzw. ein Zielvektor), Struktur (d.h. Topologie von mechanischen Komponenten, Sensoren und Aktoren), Verhalten und Parameter. Die Selbstoptimierung zeichnet sich dementsprechend durch zwei Eigenschaften aus:

- die endogene Änderung des Zielsystems aufgrund veränderter Einflüsse auf das selbstoptimierende System durch Umwelt, Benutzer und andere Systeme so wie
- die zielkonforme, selbständige Anpassung von Parametern, Verhalten und Struktur.

Die Überlagerung der regelnden Informationsverarbeitung (Controller) durch die selbstoptimierende Informationsverarbeitung (Operator) führt zu dem Ansatz des Operator-Controller-Moduls (OCM, siehe [4]). Die selbstoptimierende Informationsverarbeitung gibt der regelnden Informationsverarbeitung das Zielsystem, die Struktur, das Verhalten und die Parameter vor. Dabei sind zwei Kategorien von Wissen erforderlich:

- Das Wissen über mögliche Zielsysteme, Strukturen, Verhalten und Parameter
- Das Wissen über Auswahl und Adaptionsprozessen vorstehender Aspekte

Dieses Wissen zusammen mit der Möglichkeit der Kommunikation erlaubt es dem Operator-Controller Modul autonom und intelligent zu handeln – es entsteht der selbstoptimierende Operator-Controller-Modul-Agent (OCMA).

3 Wissensbasis von Wirkprinzipien der Selbstoptimierung

Das Wissen über mögliche Zielsysteme, Strukturen, Verhalten und Parameter so wie das Wissen über deren Auswahl und Adaption führt zum Paradigma des Wirkprinzips der Selbstoptimierung (WPso). Jeder OCM-Agent besitzt eine eigene Wissensbasis mit Wirkprinzipien der Selbstoptimierung. Die Wirkprinzipien bestehen aus Problem- und Lösungsfällen. Bild 2 veranschaulicht die Basisidee des Einsatzes dieser Wissensbasis. Durch externe Einflüsse wie Umwelt, Benutzer und andere Systeme ändert der OCM-Agent sein internes Zielsystem. Um die neuen Zielvorgaben zu erreichen, wird ein ähnlicher Problemfall aus der Wissensbasis recherchiert. Diesem historisch ähnlichen Problemfall ist ein Lösungsfall zugeordnet, der die Adaption des Systems an die Zielanforderungen beschreibt. Der selektierte Lösungsfall wird in seinen drei Komponenten Struktur, Verhalten und Parameter entsprechend den Adaptionsvorgaben an die aktuelle Situation angepasst. Auf diese Art und Weise ändert der OCM-Agent sein Verhalten gemäß den modifizierten Komponenten. Falls kein hinreichend ähnlicher Problemfall in der eigenen Wissensbasis gefunden wird, kann der OCM-Agent mit anderen Agenten kommunizieren und deren Wissen nutzen.

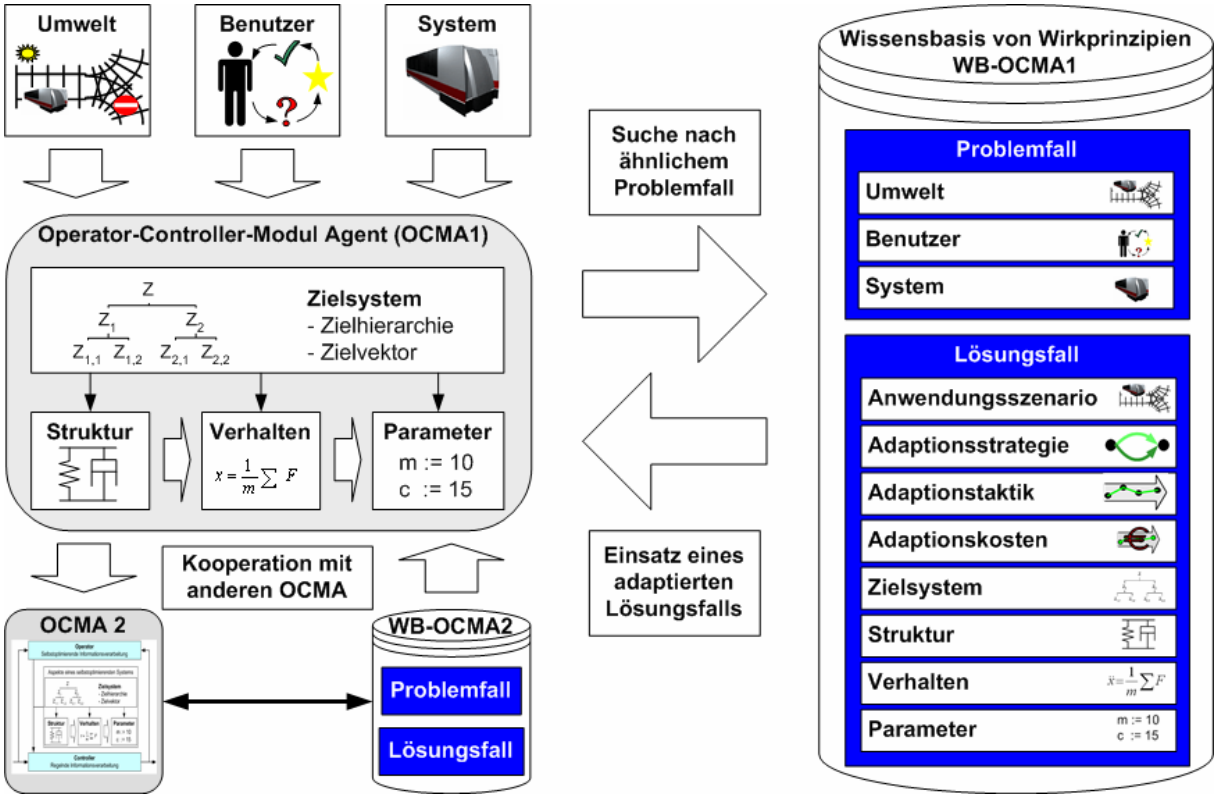

Bild 2: Die Grundidee für den Einsatz von Wissensbasen in selbstoptimierenden Systemen

Ausgehend von der Grundidee des Einsatzes von Wissensbasen für selbstoptimierende Systeme, wird für eine verhaltensbasierte Selbstoptimierung ein Meta-Modell zum Lernen aus Erfahrung benötigt. Das Meta-Modell gewährleistet den Rückgriff auf die Wissensbasis gemäß dem Prinzip erfahrungsbasierter Ähnlichkeitssuche. Lösungen ähnlicher Probleme der Vergangenheit werden in diesem Modell als relevant für die Lösung aktueller Probleme betrachtet. Das Prinzip ist auch unter dem Begriff des *Fallbasierten Schließens* [5] bekannt. Bild 3 zeigt das **Wissensmodell** eines selbstoptimierenden Systems so wie das **Vorgehensmodell** zum erfahrungsbasierten Einsatz von Wirkprinzipien der Selbstoptimierung.

Das rechts in Bild 3 gezeigte Vorgehensmodell zum erfahrungsbasierten Einsatz von Wirkprinzipien der Selbstoptimierung lehnt sich an den von [6] vorgeschlagenen *Case-Based Reasoning Cycle* an. Das Vorgehensmodell zerfällt in fünf Phasen. Zunächst identifiziert der OCM-Agent in einer Situationsanalyse das aktuelle Anwendungsszenario. Aus den Sensordaten und der Kommunikation mit anderen Systemen und dem Benutzer kompositioniert der OCM-Agent einen Problemfall bestehend aus den Teilaspekten Umwelt, Benutzer und System. Als nächstes wird mit Hilfe einer multikriteriell gewichteten Ähnlichkeitssuche der zum aktuellen Problemfall ähnlichste historische Problemfall aus der Wissensbasis der Wirkprinzipien ermittelt. Eine detaillierte Beschreibung zu verschiedenen Verfahren der multikriteriellen Ähnlichkeitssuche findet sich in [5]. Als Nächstes wird der zum ähnlichsten Problemfall zugeordnete historische Lösungsfall an die aktuelle Problemsituation angepasst. Die Wiederverwendung der historischen Lösung wird dabei über die Lösungskomponenten der Adaptionsstrategie und Adaptionstaktik gesteuert. Das Ergebnis dieser Phase besteht aus einem in seiner Struktur, seinem Verhalten und seinen Parametern adaptierter Lösungsfall. Der OCM-Agent setzt das erzeugte Verhaltensmuster in der aktuellen Situation ein und bewertet die Auswirkungen auf Umwelt, Benutzer und System. Falls die Abweichungen zwischen den erwarteten und den tatsächlichen Auswirkungen einen Schwellwert überschreiten, wird der adaptierte Lösungsfall in Richtung Minimierung der Abweichung überarbeitet. Danach wird aus dem modifizierten Lösungsfall ein neues Wirkprinzip erzeugt und als gelernter

Fall in die Wissensbasis der Wirkprinzipien eingeordnet. Das Lernen aus den mit anderen Systemen, der Umwelt und den Benutzern gemachten Erfahrungen sowie dem Lernen aus der Adaption historischer Wirkprinzipien und der Exploration wenig bekannter Anwendungsszenarien, bewirkt im Laufe der Zeit die eigentliche Selbstoptimierung des Systems.

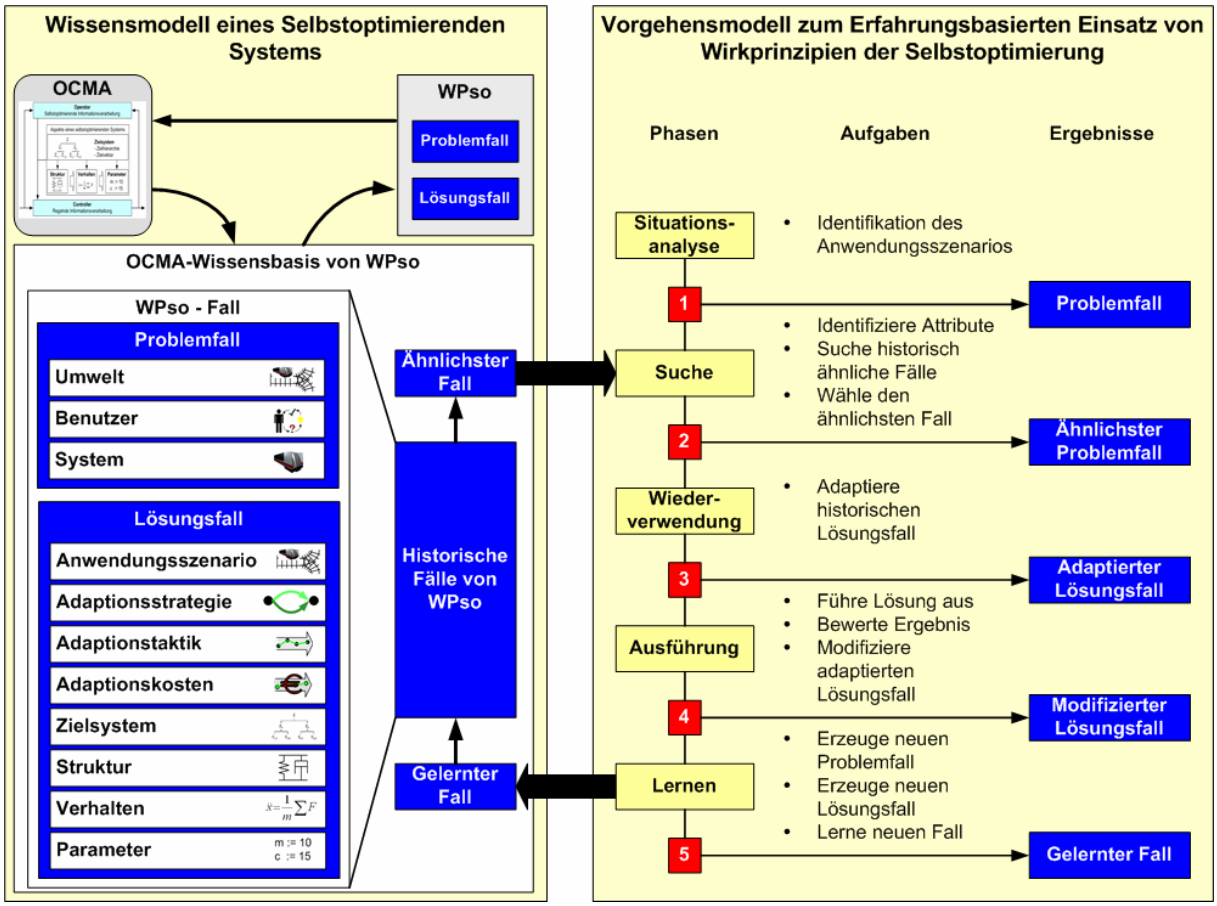

Bild 3: Wissens- und Vorgehensmodell des Einsatzes von Wirkprinzipien der Selbstoptimierung

Das Vorgehensmodell zum Lernen aus historisch gemachten Erfahrungen in Verbindung mit Wirkprinzipien der Selbstoptimierung geht über bestehende Ansätze des fallbasierten Schließens hinaus. Unser Modell umfasst den phasenübergreifenden, integrativen Einsatz in allen mechatronischen Domänen. In der Literatur findet man hingegen meist eine Fokussierung auf eine Phase, z.B. auf die Phase des Designs bei [7] oder die Anwendung auf eine Domäne, wie bei [8] auf die Software-Domäne. Des Weiteren gibt in klassischen erfahrungsbasierten Modellen das System die Adaptionsregeln für alle Anwendungsfälle vor [5]. Unser Ansatz von autonom agierenden, sich selbst optimierenden Systemelementen bedingt jedoch, dass das Adaptionswissen dezentral in den Wirkprinzipien selbst vorgehalten wird.

4 Anwendungsbeispiel Weiche befahren

Der Einsatz der vorgestellten Wirkprinzipien in selbstoptimierenden Systemen soll durch ein Beispiel aus dem Projekt „Neue Bahntechnik Paderborn“ [9] veranschaulicht werden – dem Befahren einer Weiche durch mehrere schienengebundene Shuttles. Bild 4 zeigt das Anwendungsszenario bestehend aus den Schienensegmenten *Anmeldezone*, *Verhandlungszone* und *Durchfahrtzone*, zwei Shuttles S1 und S2 mit jeweils Fahrgästen F1 und F2 sowie

einer Weichenregelung WR. Bis auf die Fahrgäste kann jedes Element als OCM-Agent mit zugehöriger Wirkprinzipien-Wissensbasis WB aufgefasst werden.

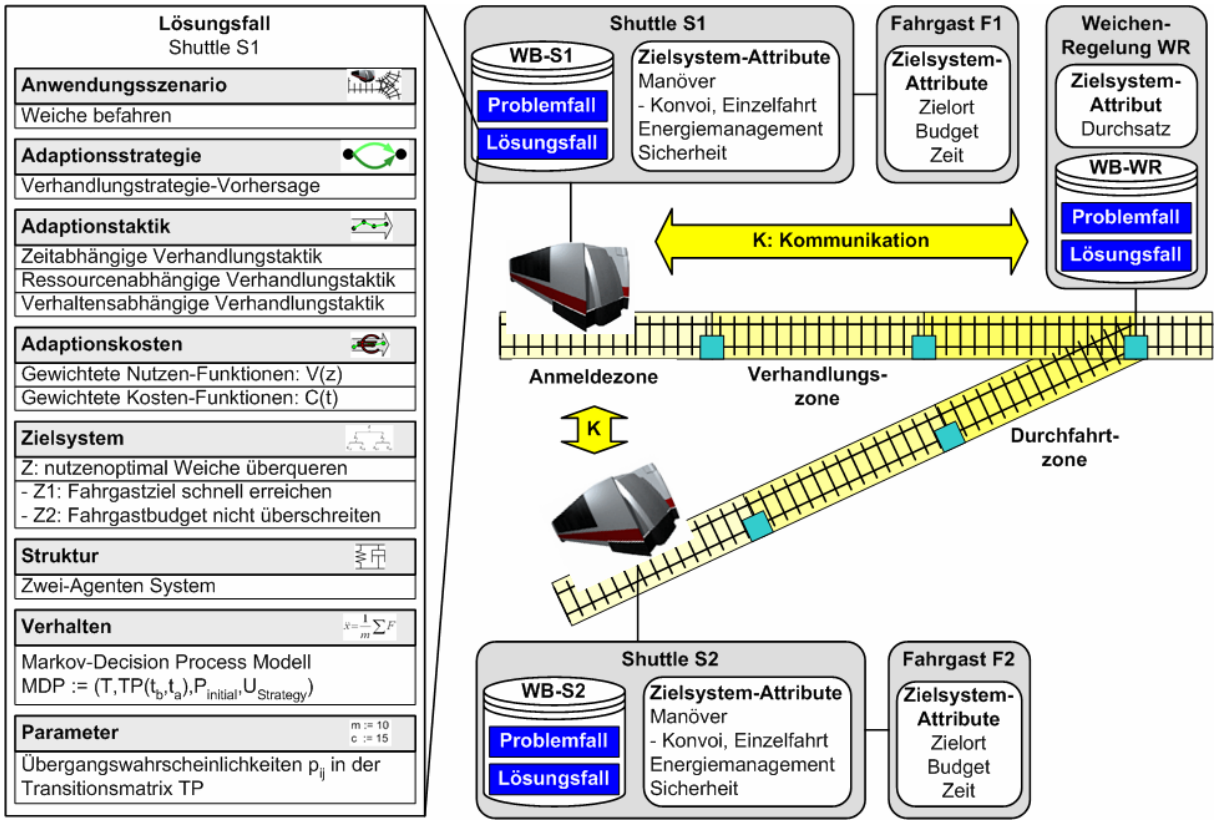

Bild 4: Anwendungsszenario Weiche Befahren

Ein typisches Zielsystem eines Shuttle-Agenten besteht aus den Attributen für ein gewünschtes Fahrverhalten im Sinne einer Einzelfahrt oder der Kopplung mit anderen Shuttles zu Konvois und dem energieoptimalen Fahren so wie dem Sicherheitsbedürfnis. Die Fahrgäste wiederum möchten innerhalb einer gewissen Zeit zu einem gewünschten Zielort durch Einsatz eines bestimmten finanziellen Budgets gelangen. Schließlich soll die Weichenregelung als oberstes Ziel einen möglichst optimalen Durchsatz an Shuttles gewährleisten. Die Shuttles und die Weiche können miteinander per Funk kommunizieren. Während sich die Shuttles in der Anmeldezone befinden, analysieren sie die Situation und selektieren ein relevantes Wirkprinzip der Selbstoptimierung aus ihrer Wissensdatenbank. Im Folgenden betrachten wir ausschließlich den Fall, dass sich beide Shuttles auf ein gemeinsames Wirkprinzip einigen konnten – hier der Festlegung der Durchfahrtreihenfolge durch Verhandlung und Ausgleichszahlung. Verhaltensstrategien bei Nicht-Einigung sind Gegenstand zukünftiger Forschung. Nähere Informationen zur Kommunikation von OCM-Agenten und der Festlegung eines gemeinsamen Wortschatzes bzw. einer gemeinsamen Ontologie finden sich in [10].

Shuttle S1 verfolge eine Adaptionsstrategie, die versucht die Verhandlungsstrategie des anderen Shuttles S2 vorherzusagen um damit die eigene Verhandlungsstrategie selbst zu optimieren. Das Vorhersagen von möglichen Verhandlungspositionen mit Hilfe so genannter *Belief-Modelle* beschreibt u.a. [11] für den Einsatz im Electronic Commerce. Je nach aktueller Situation kann S1 drei Adaptionstaktiken für den nächsten Verhandlungsschritt nutzen. Die zeitabhängige Verhandlungstaktik drängt den OCM-Agenten zum Verhandlungsabschluss. Dies ist der Fall bei Verhandlungen unter Zeitrestriktionen, wenn die Shuttles kurz vor Verlassen der Verhandlungszone stehen. Die ressourcenabhängige Verhandlungstaktik steuert das Verhaltensmuster des Agenten bei Ressourcenknappheit wie z.B. bei geringer

Energiekapazität. Die verhaltensabhängige Verhandlungstaktik steuert das eigene Verhalten in direkter Abhängigkeit vom Verhalten des anderen Shuttles. *Tit-for-Tat* ist Vertreter solch einer verhaltensabhängigen Verhandlungstaktik, bei der das Verhalten des anderen Shuttles imitiert wird. Die Adaptionskosten setzen sich aus gewichteten Kosten- und Nutzenfunktionen zusammen. Die Kosten entstehen in diesem Szenario bei der Verzögerung des Verhandlungsverlaufes. Der Nutzen bezieht sich auf die Wichtigkeit, die der OCM-Agent dem Erreichen eines spezifischen Zieles zuordnet. Das Zielsystem gibt das nutzenoptimale Überqueren der Weiche aus den untergeordneten Teilzielen des schnellen Erreichens des Fahrgastzieles und der Einhaltung des vorgegebenen Fahrgastbudgets vor. Die Struktur des Gesamtsystems setzt sich aus einem Zwei-Agenten Szenario – Shuttle-Agent S1 und Shuttle-Agent S2 – so wie deren Interaktion zusammen. Das Verhalten im Sinne der Vorhersage der Verhandlungstaktik des jeweils anderen Shuttles folgt einem *Markov-Decision Prozessmodell* MDP := (T, TP(t_{S2},t_{S1}), $P_{initial}$, $U_{Strategy}$). Der MDP besteht aus dem Tupel der Menge verfügbarer Adaptionstaktiken T und der Transitionsmatrix TP bedingter Übergangswahrscheinlichkeiten – Shuttle-Agent S2 ändert seine Taktik t_{S2} unter der Annahme, dass Shuttle-Agent S1 eine Taktik t_{S1} wählt. Ausserdem modelliert der MDP eine a-priori Wahrscheinlichkeitsverteilung der Menge initialer Verhandlungstaktiken $P_{initial}$ und ein Nutzenmodell $U_{Strategy}$, das einen erwarteten Nutzen E(U) einer Strategie zuweist, die ein Agent aufgrund der Strategiewahl des anderen Agenten wählt. Die Parameter bestehen in dem MDP-Modell aus den Übergangswahrscheinlichkeiten p_{ij} der Transitionsmatrix TP = [p_{ij}]. Dabei spiegelt p_{ij} die bedingte Wahrscheinlichkeit wider, dass Shuttle-Agent S1 zum Zeitpunkt t+2 zu der Taktik t_j wechselt, falls Shuttle-Agent S2 zum Zeitpunkt t die Taktik t_i gewählt hatte.

Die Umsetzung des Markov-Decision Prozessmodells repräsentiert ein Lösungselement für das Wirkprinzip des verhandlungsbasierten Weiche Befahrens in der Form eines konkret ausgeprägten Algorithmus mit zugehöriger Parametrisierung. Die gemachten Erfahrungen in Form der Transitionsmatrix TP und der eingesetzten Adaptionsstrategien und -taktiken werden in der Wissensbasis der Wirkprinzipien jedes beteiligten Agenten gespeichert. Die Selbstoptimierung der beteiligten OCM-Agenten wird durch die mit der Zeit immer genaueren Vorhersagen des Verhandlungsverhaltens des jeweils anderen Agenten erreicht.

5 Literatur

[1] Gausemeier, J.: From Mechatronics to Self-Optimization, 20th CAD-FEM Users Meeting 2002, International Congress on FEM Technology, Oct. 9-11, 2002, Friedrichshafen

[2] Pahl G. Beitz W.: Konstruktionslehre – Methoden und Anwendungen, 5. Auflage, Springer-Verlag, Berlin 2003

[3] SFB 614: Einrichtungsantrag für den Sonderforschungsbereich 1799 (ab 1. Juli 2002: 614) „Selbstoptimierende Systeme des Maschinenbaus“, Universität Paderborn, 2001

[4] Naumann, R.: Modellierung und Verarbeitung vernetzter intelligenter mechatronischer Systeme, Dissertation im Fachbereich 10 Maschinentechnik der Universität Paderborn, VDI Verlag, Reihe 20, Nummer 318, Düsseldorf, 2000

[5] Bergmann, R.: Experience Management – Foundations, Development Methodologies, and Internet-Based Applications, Springer Lecture Notes in Artificial Intelligence 2432, 2002

[6] Aamodt, A, Plaza, E.: Case-Based Reasing: Foundational issues, methodological variations, and system approaches, AI Communications, Vol. 7, No. 1, pp. 39-59, 1994

[7] Stein, B., Vier, E.: An Approach to Formulate and to Process Design Knowledge in Fluidics, In: Mastorakis, N.E. (ed.) Recent Advantages in Information Science and Technology, pp. 237-242, Oct. 1998, World Scientific Publishing Co. Pte. Ltd.

[8] Smyth, B, Keane, M.T., Cunningham, P.: Hierarchical Case-Based Reasoning Integrating Case-Based and Decompositional Problem-Solving Techniques for Plant-Control Software Design, In: IEEE Transactions on Knowledge and Data Engineering, Vol. 13, No. 5, pp. 793-812, Sept./Oct. 2001

[9] Neue Bahntechnik Paderborn, Homepage: http://nbp-www.upb.de, 2003

[10] Scheideler, P., Schmidt, A.: Entwicklung einer Modellwelt für ein dezentrales, intelligentes, mechatronisches System, In: Gausemeier, J., Lückel, J., Wallaschek, J. (Hrsg.) Intelligente Mechatronische Systeme, HNI-Verlagsschriftenreihe Band 122, S. 81-91, 2003

[11] Teuteberg, F., Kurbel, K.: Anticipating Agents' Negotation Strategies in an E-marketplace Using Belief Models, In: Abramovicz, W. (ed.): Proceeding of the 5th International Conference on Business Information Systems (BIS2002), pp. 91-100, Posen, Polen, 2002

Prof. Dr.-Ing. Jürgen Gausemeier
Dipl.-Wirt.-Ing. Andreas Schmidt
Heinz Nixdorf Institut, Universität Paderborn
Fürstenallee 11, D-33102 Paderborn
Tel: +49-5251-60-6262
Fax: +49-5251-60-6268
Email: Juergen.Gausemeier@hni.uni-paderborn.de
Andreas.Schmidt@hni.uni-paderborn.de
URL: http://wwwhni.upb.de/rip