Flexible aufgabenorientierte Robotersteuerungen auf Basis der System Entity Structure / Model Base

Tobias Schwatinski, Thorsten Pawletta, Sven Pawletta

Hochschule Wismar – University of Applied Sciences: Technology, Business and Design
Research Group Computational Engineering and Automation

{tobias.schwatinski, thorsten.pawletta, sven.pawletta} @ hs-wismar.de

Zusammenfassung

Es wird eine Methode zur systematischen Entwicklung flexibler aufgabenorientierter Robotersteuerungen (FAR) auf Basis der System Entity Structure (SES) eingeführt. Aufgabenorientierte Robotersteuerungen basieren auf einer Komposition von Basisaufgaben zur Umsetzung einer gegebenen Zielstellung. Bei flexiblen aufgabenorientierten Steuerungen ist die Komposition der Aufgaben nicht fest vorgegeben, sondern ergibt sich unter Einhaltung vorgegebener Zwangsfolgen erst während des Betriebes auf Basis der aktuellen Pozesszustände. Die System Entity Structure ist eine Ontologie, welche beliebige Gegenstandsbereiche hierarchisch repräsentieren kann. Es wird gezeigt, wie die SES zur Spezifikation von FAR für kooperierende Roboter genutzt und aufbauend auf eine Modellbasis (MB) automatisch Steuerungsvarianten abgeleitet und ausgeführt werden können.

1 Einleitung

Flexible aufgabenorientierte Robotersteuerungen bestehen aus einzelnen Aufgaben, die unter Einhaltung von Zwangsfolgen, flexibel zur Erledigung einer Zielstellung komponiert werden. Konkrete Aufgabenfolgen ergeben sich erst während des Betriebs aus den aktuellen Prozesszuständen, beziehungsweise aus Ergebnissen prädiktiver Prozesssimulationen. Damit gehören sie nach [7] zur Klasse der Intelligenten Robotersteuerungen und zählen bezüglich ihrer Umsetzung aufgrund der Anforderungen und Komplexität gemäß [1] zur Steuerungsentwicklung „im Großen“. Daraus folgt, dass die Programmierung derartiger Robotersteuerungen als systematischer Entwicklungsprozess gestaltet werden muss.

Im Rahmen dieser Arbeit wird eine Methode (FAR/SES) zur systematischen Entwicklung flexibler aufgabenorientierter Robotersteuerungen (FAR) auf Basis der System Entity Structure (SES) und Model Base (MB) eingeführt, welche für kooperierende Robotersteuerungen Verwendung finden kann. Die System Entity Structure stellt das zentrale Element der FAR/SES Methode dar. Die SES wurde von Rozenblit und Zeigler 1986/89

eingeführt und ist bis zum Data Engineering [3] kontinuierlich weiterentwickelt worden. Sie ist eine Ontologie, die einen beliebigen Gegenstandsbereich hierarchisch repräsentiert und wird vielfach zur deklarativen Beschreibung von Systemvarianten innerhalb der Simulationstechnik verwendet. In diesem Fall wird die SES zur deklarativen Beschreibung von Industrierobotersteuerungen verwendet. Die modulare und deklarative Beschreibung der gesamten Steuerung einschließlich der zu steuernden Prozesse mit einer SES unterstützt neben einer systematischen Entwicklung auch die Wiederverwendbarkeit, Anpassung und Wartung der Steuerung.
Weiterhin basiert die FAR/SES Methode auf dem Simulationsmodellbasierten Steuerungsansatz (SBC) nach [2] und erlaubt eine sukzessive Entwicklung vom Entwurf bis zur Realisierung mit Simulationsmodellen in einer homogenen Entwicklungsumgebung. Der SBC stellt eine konkrete Realisierung des Rapid Control Prototypings (RCP) gemäß [4] dar.
Es werden die Grundlagen der SES und des SBC kurz eingeführt. Anschließend wird deren gemeinsame Verwendung zur deklarativen Beschreibung von kooperierenden Robotersteuerungen nach der FAR/SES Methode anhand eines Beispiels erörtert. Nachfolgend wird gezeigt, wie während der Steuerungsausführung, aufbauend auf eine Modellbasis (MB) aus der deklarativen Steuerungsbeschreibung automatisch Steuerungsprogramme adaptiert werden. Abschließend wird ein kurzer Erfahrungsbericht zur Umsetzung einer realen Roboterapplikation mittels der FAR/SES Methode gegeben.

2 Grundlagen der System Entity Structure und des Simulationsmodellbasierten Steuerungsansatzes

2.1 Die System Entity Structure

Die System Entity Structure (SES) ist eine Ontologie. Sie bildet einen Baum, deren Knoten nach [3] in vier unterschiedlichen Ausprägungen auftreten können. Die Ausprägungen sind Entity-, Aspect-, Multiple-Aspect- und Specialization-Knoten. Die prinzipielle Abfolge der Knoten innerhalb einer SES ist in Abb.1 (a) dargestellt.
Entity-Knoten repräsentieren die Elemente einer realen oder imaginären Welt. Aspect-Knoten dienen der Zerlegung eines Entity-Knotens in feiner aufgelöste Strukturen. Multiple-Aspect-Knoten definieren die Vielfachheit des nachfolgenden Entity-Knotens und Specialization-Knoten repräsentieren Kategorien oder Familien bestimmter Ausprägungen eines Entity-Knotens. Darüber hinaus können jedem Knoten Attribute, einschließlich ihrer Definitionsbereiche, zugewiesen werden. Abbildung 1 (b) zeigt beispielhaft die SES zur Spezifikation verschiedener Automodelle. Jede Entity Auto besteht entsprechend dem Aspect-Knoten aus den Entities Abgas, Motor, Räder und Chassis. Die Entity Motor wird dabei entweder spezialisiert zu der Entity Diesel oder der Entity Otto. Darüber hinaus wird der Entity Otto ein Attribut Takt zugewiesen, welches den Wert 2 oder 4 annehmen kann. Weiterhin unterstützt die SES die Spezifikation von Zwangsfolgen bei der Auswahl von Alternativen. Im Beispiel wird jeder Entity Diesel die Entity Partikelfilter und jeder Entity Otto die Entity Katalysator zwangsweise zugeordnet. Auf die Entity Räder folgt ein Multiple-

Aspect-Knoten mit dem Parameter *Vielfachheit* = 4. Deshalb wird dieser Knoten durch vier M&S Entities aufgelöst.

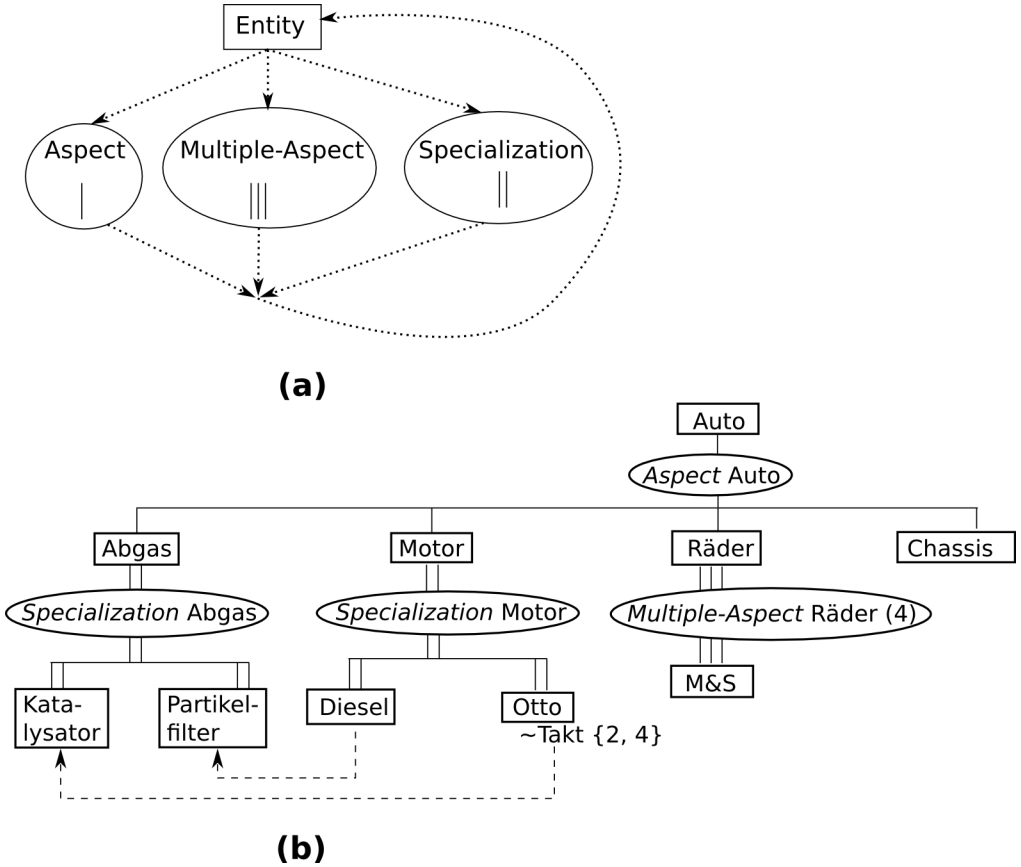

Abbildung 1: Aufbau einer SES

Die SES in Abbildung 1 (b) beschreibt demnach die drei verschiedenen Automodelle:

- Modell: Dieselmotor, Partikelfilter, 4 M&S Reifen, Chassis
- Modell: 2 Takt Otto Motor, Katalysator, 4 M&S Reifen, Chassis
- Modell: 4 Takt Otto Motor, Katalysator, 4 M&S Reifen, Chassis

Auf diese Weise lassen sich mit einer SES eine Vielzahl von verschiedenen Ausprägungen eines beliebigen Systems anschaulich darstellen. Ursprünglich entwickelt wurde die SES zur deklarativen Beschreibung von Modellen in der Simulationstechnik. Wenn die SES mit einer Modellbasis (MB) kombiniert wird, lassen sich aus ihr mit Hilfe eines Modellgenerators vollständige Simulationsmodelle ableiten. Diese erweiterte Form der SES, SES/MB genannt, weist jedem Blattknoten der Baumstruktur genau ein Softwaremodul der Modellbasis zu. Dazu ist es notwendig, dass an den Aspect-Knoten die entsprechenden Kopplungsbeziehungen der Entities spezifiziert werden.

2.2 Der Simulationsmodellbasierte Steuerungsansatz

Der Simulationsmodellbasierte Steuerungsansatz (SBC) nach [2] ist eine spezielle Form des Software in the Loop (SiL) Prinzips [4] und ermöglicht die durchgängige Nutzung von Simulationsmodellen während der gesamten Steuerungsentwicklung. Simulationsmodelle aus

der Entwurfsphase werden in der Automatisierungsphase schrittweise erweitert und unter Verwendung eines Prozessinterfaces unmittelbar als Steuerungssoftware eingesetzt. Dieser Ansatz erlaubt es den Entwicklungsrechner oder einen Industrie-PC unmittelbar zum Regeln/Steuern realer Prozesse einzusetzen. Dafür muss die Ausführung der Simulationsmodelle allerdings mit der Echtzeit synchronisiert werden. Der Entwicklungsprozess von Steuerungen nach dem SBC ist schematisch in Abbildung 2 dargestellt. Der SBC fordert, dass die Steuerungssoftware aus einem Steuerungsmodell, einem Interface und gegebenenfalls einem Prozessmodell besteht. Das Interface stellt die Schnittstelle zwischen dem realen zu steuernden Prozess und der Steuerungssoftware dar. Diese Form der Kommunikation mit einem realen Prozess wird als implizite Codegenerierung bezeichnet. Im Prozessmodell sind die Elemente, die Zustände und das Verhalten des realen Prozesses modelliert. Zudem kann ein Prozessmodell, welches bereits in der Automatisierungsphase genutzt wird, in die Steuerungssoftware integriert werden und als Prozessbeobachter verwendet werden. Dieses Vorgehen kann die Qualität von Steuerungen erhöhen, z.B. durch die Berechnung zusätzlicher oder nicht messbarer Zustandsgrößen. Das Steuerungsmodell bildet die gesamte Steuerungslogik ab. Die einzelnen Ebenen des SBC kommunizieren miteinander, wie in Abbildung 2 dargestellt.

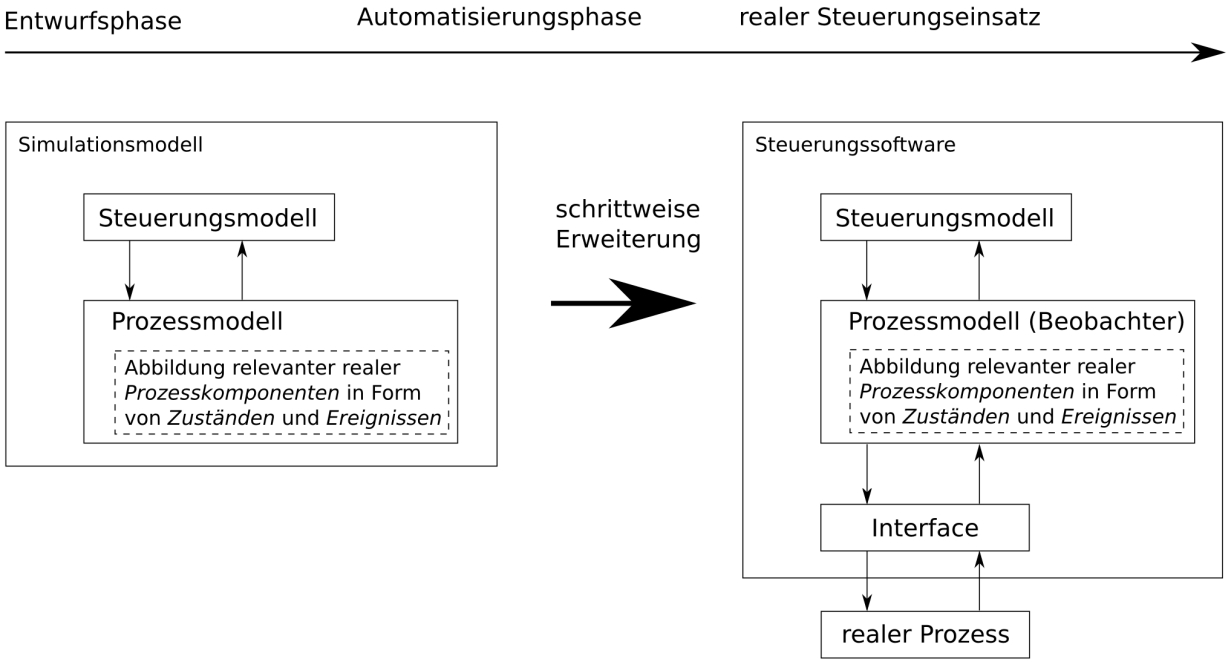

Abbildung 2: Simulationsmodellbasierter Steuerungsansatz (SBC)

3 Entwicklung von Robotersteuerungen auf Basis einer deklarativen Steuerungsspezifikation

3.1 Integration von deklarativer Beschreibung und Simulationsmodellbasiertem Steuerungsansatz

Die Anwendung des SBC ermöglicht eine effektive Entwicklung von Robotersteuerungen mit Simulationsmodellen, beginnend in der Entwicklungsphase bis hin zum realen

Steuerungseinsatz. Die SES/MB erlaubt die systematische und deklarative Darstellung von beliebigen Ausprägungen eines dynamischen Systems in einer Baumstruktur und die automatische Generierung von Software aus vordefinierten parametrierbaren Modulen. Nachfolgend wird gezeigt, wie durch eine Kombination der beiden Ansätze eine deklarative Beschreibung von aufgabenorientierten Robotersteuerungen erfolgen kann und wie auf dieser Basis flexible Steuerungen realisiert werden.

Der SBC unterstützt nach [2] die Umsetzung aufgabenorientierter Steuerungen. Vordefinierte Aufgabenmodule werden in einem Steuerungsmodell entsprechend der vorgegebenen Zielstellung komponiert und parametriert. Dabei beinhalten die einzelnen Aufgabenmodule die vollständigen Steuerungsanweisungen und Reaktionen auf Zustandswerte des realen Systems, beziehungsweise des Prozessmodells. Dieses Prinzip zeigt Abbildung 3 schematisch am Steuerungsmodell eines einfachen Beispiels.

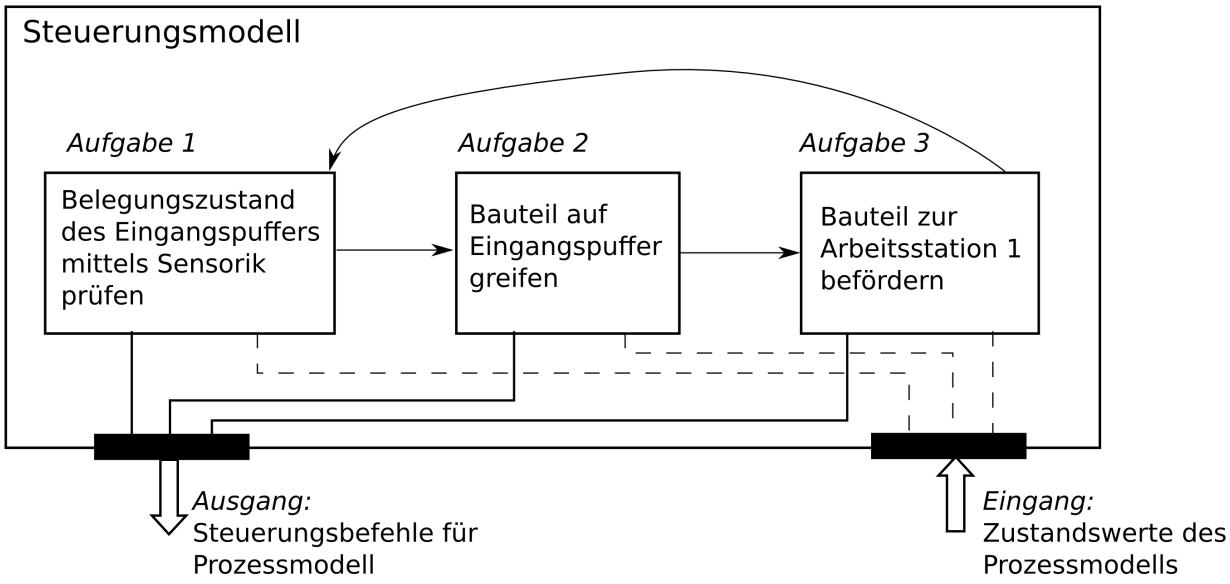

Abbildung 3: Kopplung von Aufgabenmodulen im Steuerungsmodell nach dem SBC

Abbildung 3 zeigt, dass das Steuerungsmodell eine feste Abfolge von Aufgaben implementiert. Jegliche Flexibilität der Steuerung muss damit entweder in den einzelnen Aufgabenmodulen oder in deren geschickter Kopplung liegen. Insbesondere bei komplexen und flexiblen Roboterapplikationen kommt es zu vielfältigen Wechselwirkungen zwischen den Modulen im Steuerungsmodell untereinander, als auch dem Steuerungsmodell und dem Prozessmodell in der Gesamtheit. Die hohe Komplexität einer derartigen Steuerung liegt darin begründet, dass der SBC zwar ein strukturiertes Modellieren in einem Softwaresystem fördert, aber starr definiert, wie einzelne Aufgaben zueinander in Beziehung stehen.

Die deklarative Beschreibung von Steuerungen soll diesen Nachteil innerhalb des SBC entgegenwirken und somit den gesamten Steuerungsentwurf flexibler und anpassungsfähiger gestalten und damit die Grundlage für flexible aufgabenorientierte Robotersteuerungen schaffen. Die deklarative Steuerungsbeschreibung soll auf Basis des SES Ansatzes erfolgen. Die wichtigsten Elemente und Wechselwirkungen einer flexiblen aufgabenorientierten

Robotersteuerung nach der FAR/SES Methode sind in Abbildung 4 dargestellt. Die SES spezifiziert die Strukturen und Parameter des gesamten Systems, d.h. sowohl die flexible Steuerung als auch die unterlagerten Prozesse und Prozessschnittstellen. Da die Syntax und

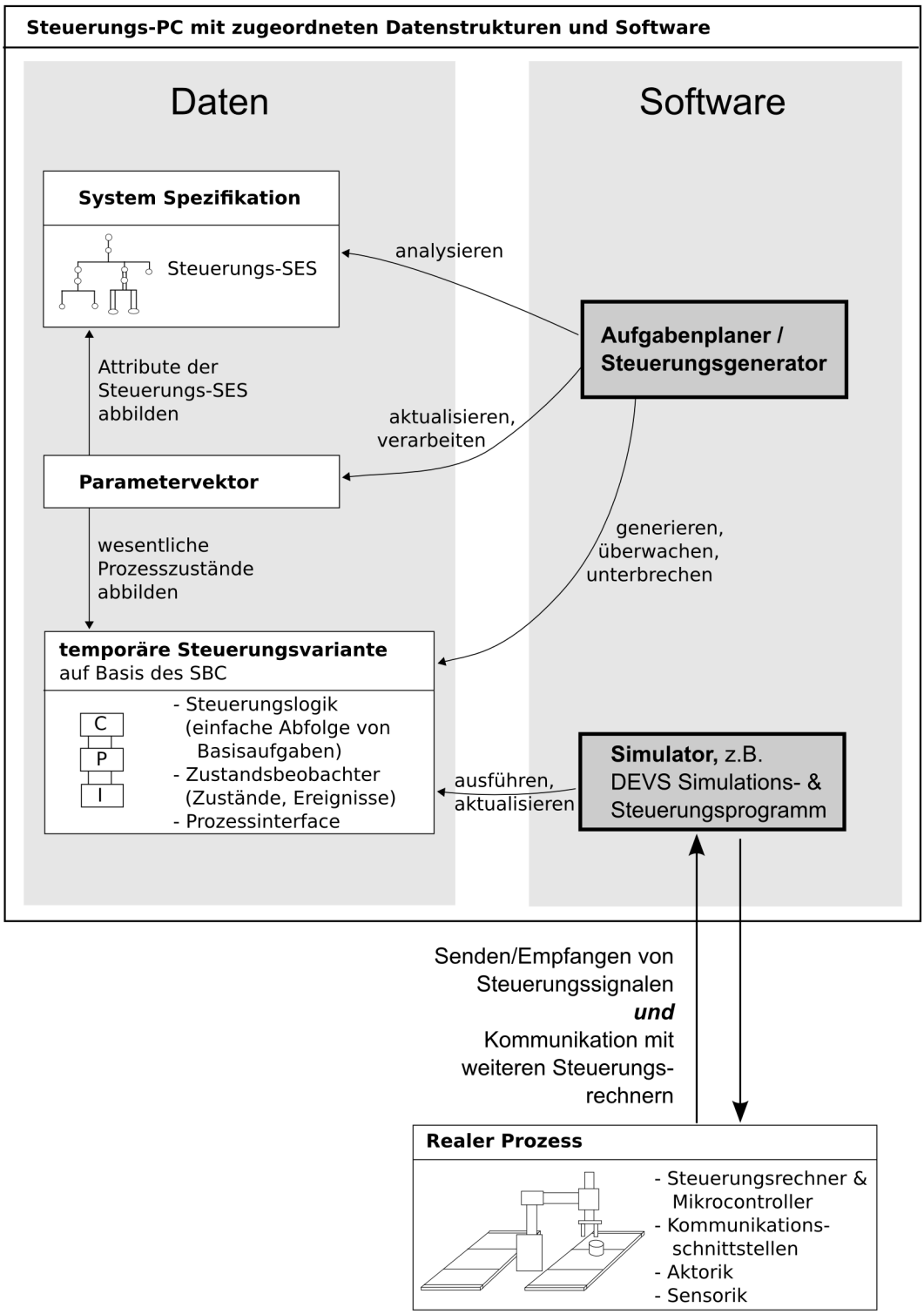

Abbildung 4: Elemente und Wechselwirkungen einer flexiblen aufgabenorientierten Robotersteuerung nach der FAR/SES Methode

Semantik der SES gegenüber den originären Definitionen in [3] erweitert wurden, wird diese als Steuerungs-SES bezeichnet. Zur Vereinfachung wird nachfolgend dennoch oft der Term

SES verwendet. Die Steuerungs-SES repräsentiert die gesamte Menge möglicher Steuerungsstrukturen in Form zu komponierender Basisaufgaben. Neben den Steuerungsstrukturen sind auch die möglichen Parametrierungen spezifiziert. Eine parametrierte Steuerungsstruktur wird nachfolgend als Steuerungsvariante bezeichnet. Die Verbindung zwischen einer konkreten Steuerungsvariante und der Steuerungs-SES bildet der Parametervektor. Dieser Vektor bildet genau die Zustände einer konkreten Steuerungsvariante ab welche auch Attribute (Parametrierungen) der Steuerungs-SES sind. Die Synthese und Generierung einer konkreten, oftmals nur temporären, Steuerungsvariante erfolgt durch den Aufgabenplaner/Steuerungsgenerator (AP/SG). Dazu verarbeitet der AP/SG den Parametervektor und bildet dessen Elemente bei der Analyse der Steuerungs-SES auf sämtliche Attribute ab. Dadurch kann zu jedem Parametervektor eine eindeutige Steuerungsvariante aus der Steuerungs-SES generiert werden. Die Ausführung der aktuellen Steuerungsvariante erfolgt auf einem Steuerungs-PC in einem geeigneten Simulator. Dieser kommuniziert mit der Sensorik und Aktorik des realen Prozesses sowie anderen Steuerungsrechnern und aktualisiert dementsprechend Zustände und Ereignisse der aktuellen Steuerungsvariante fortwährend. Während der Ausführung einer Steuerungsvariante durch den Simulator überwacht der AP/SG sämtliche Zustandsänderungen und Ereignisse. Daneben aktualisiert der AP/SG den Parametervektor, indem die wesentlichen Zustände der Steuerungsvariante mit dem Parametervektor fortwährend abgeglichen werden. Die aktuelle Steuerung wird durch den AP/SG unterbrochen, sobald definierte Ereignisse, z.B. Sensorsignale, eintreten. Dies führt zur zur Synthese und Generierung einer neuen (temporären) Steuerungsvariante. Dieser Vorgang wird iterativ fortgeführt, bis ein definiertes Abbruchkriterium eintritt. In den folgenden Abschnitten werden insbesondere die deklarative Beschreibung einer Robotersteuerung und die Funktionsweise des AP/SG diskutiert.

3.2 Deklarative Spezifikation von Robotersteuerungen

Im Rahmen dieses Beitrags wird die SES zur Spezifikation von flexiblen Industrierobotersteuerungen eingesetzt. Als beispielhafte Problemstellung soll eine Sortieraufgabe dienen, bei der zwei Roboter den Belegungszustand von zwei Puffern in einen vom Benutzer definierten Zielzustand überführen. Jedem Roboter ist genau ein Puffer zugeordnet, welcher ausschließlich durch diesen bedient wird. Weiterhin können sich die Roboter gegenseitig Objekte übergeben. Die Steuerung bestimmt automatisch die günstigste Sortierfolge und lässt diese durch die Roboter ausführen. Die Aufgabenstellung ist kooperativ. Jeder Roboter muss zielfremde Objekte in seinem Puffer zwischenspeichern, damit der jeweils andere Roboter notwendige Umsortierungen durchführen kann. Da die Anzahl der zu sortierenden Objekte deutlich größer als die Anzahl der Ablageplätze in den Puffern ist, müssen die Objekte in den Puffern gestapelt werden. Die Objektübergabe erfolgt unmittelbar zwischen den Robotern. Eine vereinfachte Steuerungs-SES einer derartigen kooperativen Robotersteuerung mit zwei Robotern zeigt Abbildung 5. Der Übersicht wegen, beinhaltet die dargestellte SES lediglich eine kleine Auswahl realisierbarer Basisaufgaben.

Darüber hinaus zeigt Abbildung 6 zwei aus dieser SES abgeleitete (temporäre) Steuerungsvarianten.

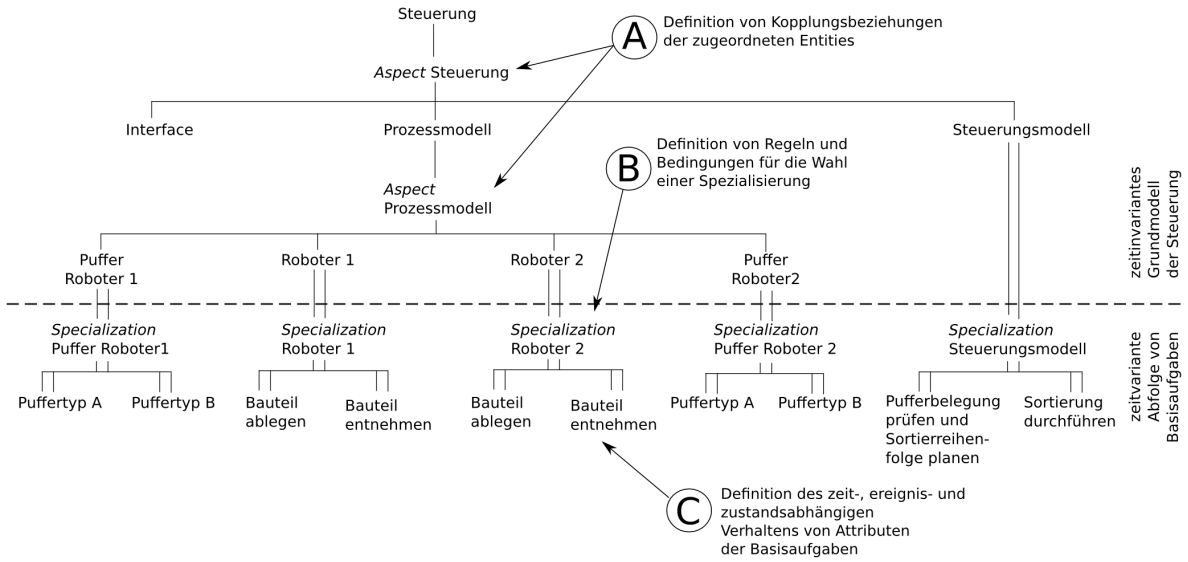

Abbildung 5: Deklarative Beschreibung einer kooperativen Robotersteuerung mittels der SES

Die dargestellte Steuerungs-SES setzt sich aus zwei Bestandteilen zusammen. Im oberen Bereich ist das zeitinvariante Grundmodell der Steuerung spezifiziert, welches in Anlehnung an den SBC Steuerungsmodelle, Prozessmodelle und Interfaces definiert. Die Gesamtaufgabe der zu realisierenden Robotersteuerung ist in Teilaufgaben strukturiert, welche im unteren Bereich, in Form von zeitvarianten alternativen und kooperierenden Aufgabenmodulen spezifiziert sind. Im Gegensatz zum SBC sind bei der FAR/SES Methode sowohl das Prozessmodell als auch das Steuerungsmodell modular aufgebaut. Diese Modularisierung stellt einen großen Vorteil gegenüber dem SBC dar, da sowohl die Komponenten des Prozessmodells als auch des Steuerungsmodells nicht mehr starr miteinander beziehungsweise untereinander gekoppelt sind. So muss ein konkretes Prozessmodell nicht mehr alle denkbar möglichen Materialflüsse, Zustände und Schnittstellen zum Steuerungsmodell beinhalten, sondern kann passend zu jedem Steuerungsmodell adaptiert werden. Da jede Komponente eines adaptierten Prozessmodells lediglich bestimmte Zustandsfolgen und Ereignisse spezifiziert, wird ihr konkretes Verhalten auch in Aufgabenmodulen beziehungsweise Basisaufgaben implementiert. Darüber hinaus ist ebenso die Abfolge der Basisaufgaben des Steuerungsmodells nicht mehr starr definiert. Diese Strukturierung bildet im Folgenden die Grundlage für einen deklarativen baukastenorientierten Steuerungsentwurf bis hin zur Steuerungsausführung.

Die Blattknoten (C) der Steuerungs-SES in Abbildung 5 stellen nicht weiter zerlegbare (atomare) Basisaufgaben dar, welche als parametrierbare Softwaremodule realisiert und in einer Modellbasis (MB) abgelegt sind. Jeder Basisaufgabe können Attribute zugewiesen werden. Die Wertzuweisung an Attribute ist zeit-, ereignis- und zustandsabhängig und wird durch entsprechende logische Ausdrücke als Eigenschaft eines Knotens spezifiziert. Zusätzlich definiert jeder Aspect-Knoten (A) die Kopplungsbeziehungen der Entity-Knoten

der jeweils nächsten Ebene. Weiterhin definiert jeder Specialization-Knoten (B) Regeln und Bedingungen, welche die Auswahl der zugeordneten Basisaufgaben spezifizieren.
Durch Vorgabe eines Parametervektors, welcher aktuelle Werte für die Attribute der Steuerungs-SES definiert, werden aus der SES in Abbildung 5 temporär gültige Steuerungsvarianten synthetisiert. Dieser Vorgang wird nach [3] *Prunen* genannt. Der Aufbau der abgeleiteten temporär gültigen Steuerung basiert stets auf dem zeitinvarianten Grundmodell der SES Spezifikation, wobei jede Entity des zeitinvarianten Grundmodells durch eine konkrete zeitvariante Basisaufgabe substituiert wird. Die Auswahl der Basisaufgabe erfolgt entsprechend des vorgegebenen Parametervektors an den Specialization-Knoten. Das grundlegende Vorgehen der Synthese konkreter Steuerungsvarianten ist in Abbildung 6, ausgehend von der Steuerungs-SES in Abbildung 5, schematisch dargestellt.

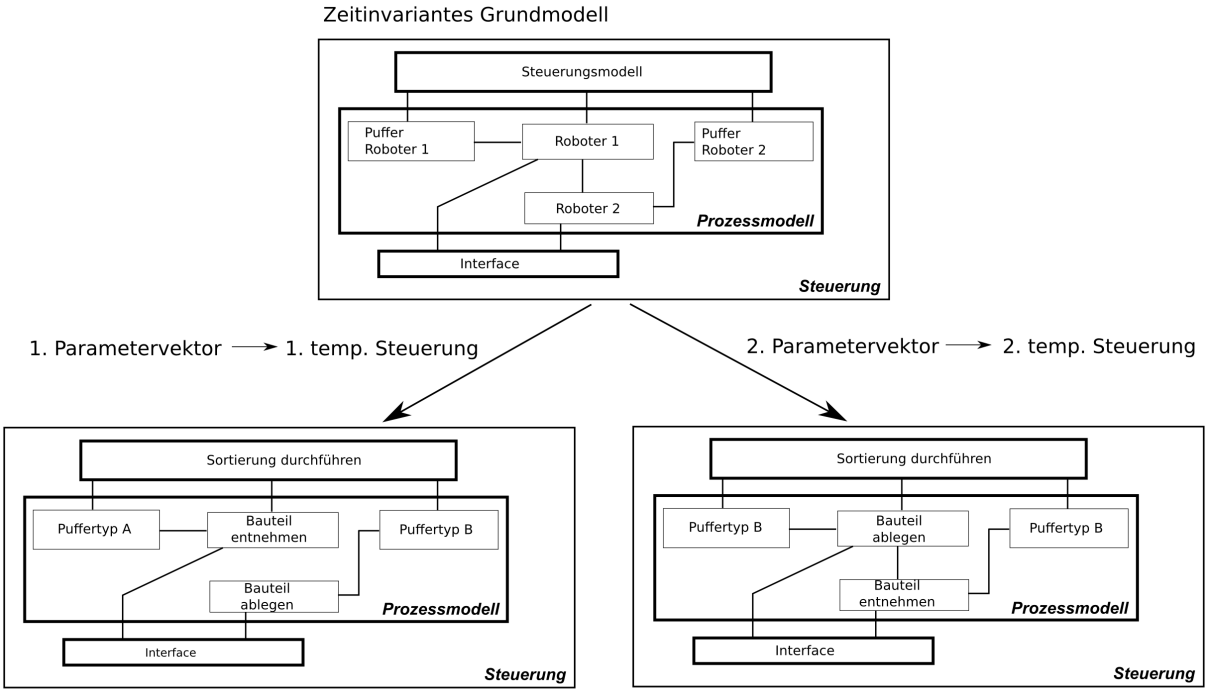

Abbildung 6: Ableitung von temporären Steuerungsvarianten aus der SES

3.3 Ableitung von Aufgabenfolgen

Beide Steuerungsvarianten in Abbildung 6 realisieren jeweils eine Teilmenge des abzuarbeitenden Aufgabenspektrums der aufgezeigten Roboterapplikation. Die tatsächliche Abfolge von Basisaufgaben (C) wird erst zur Laufzeit der Steuerung auf Basis der aktuellen Prozesszustände ermittelt. Die Flexibilität der Steuerung folgt aus der iterativen Synthese und Generierung von temporär gültigen Steuerungsvarianten. Voraussetzung dafür ist eine angemessene Zerlegung der gesamten Steuerung in Teilaufgaben. Die Spezifikation einzelner Aufgaben und die Ableitung von Aufgabenfolgen soll nachfolgend am Beispiel des Roboters 1 der Steuerungs-SES in Abbildung 5 dargestellt werden. Dazu zeigt Abbildung 7 eine Detaillierung der Steuerungs-SES für Roboter 1. Im zeitinvarianten Grundmodell der Entity *Roboter 1* werden deren wesentliche Prozesszustände und Prozessereignisse, einschließlich deren Wertebereiche, in Form von Attributen definiert. Diese und andere Attribute von

Prozesskomponenten können im zeitvarianten Grundmodell verarbeitet werden. Innerhalb der Spezialisierung der Entity *Roboter 1* ist die Auswahl von Basisaufgaben während der Steuerungssynthese, nach [3] *Prune*-Prozess genannt, definiert. Die Basisaufgabe *Bauteil entnehmen* wird bei der Synthese ausgewählt, wenn der Roboter 1 nicht belegt ist (R1_Obj == *frei*) und sich im Eingangspuffer Objekte zur Entnahme (Puffer ≠ *leer*) befinden. Jede Basisaufgabe definiert in dieser Form mit Attributen und Regeln ein zeit-, zustands- und ereignisabhängiges Verhalten. Die Werte von Attributen können sich während der Steuerungsausführung verändern und führen bei nachfolgenden Auswertungen der SES zu neuen Steuerungsvarianten. Die aktuellen Attributwerte werden vor jeder Steuerungssynthese als aktueller Parametervektor zur Verfügung gestellt. Wenn z.B. die Basisaufgabe *Bauteil entnehmen* Bestandteil der aktuellen Steuerung ist und ein Objekt von Roboter 1 im Eingangspuffer erkannt und gegriffen wurde (R1_event == *Objekt erkannt*), dann wird die Bezeichnung des Objektes im Attribut R1_Obj abgespeichert und eine neue Steuerungssynthese aktiviert, bei welcher die Basisaufgabe Bauteil ablegen ausgewählt wird, da die Bedingung R1_Obj ≠ *frei* erfüllt ist. Die Aktivierung einer neuen Steuerungssynthese nach Aufnahme eines Objektes durch den Roboter ist in der Steuerungs-SES durch den Term „R1_Obj = {?} AND PRUNE | R1_event == *Objekt erkannt*“ definiert. Auf diese Weise ergeben sich die konkreten Aufgabenfolgen prozessabhängig zur Laufzeit der Steuerung.

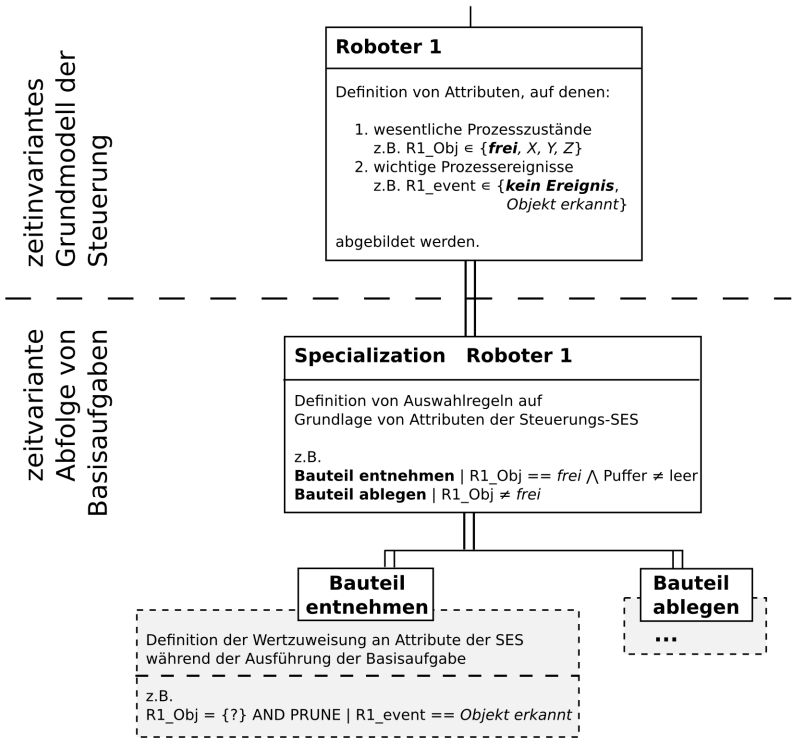

Abbildung 7: Erweiterte Spezifikation der Steuerungs-SES für Roboter 1

4 Automatische Generierung temporärer Steuerungsprogramme

Abbildung 8 zeigt schematisch die automatische Generierung temporärer Steuerungsprogramme. Ausgangspunkt einer Steuerungssynthese ist immer ein aktueller Parametervektor gemäß Abschnitt 3.2 und 3.3, der alle Attribute der Steuerungs-SES abbildet.

Im ersten Schritt analysiert der Aufgabenplaner die Steuerungs-SES unter Verwendung des aktuellen Parametervektors und synthetisiert eine konkrete Steuerungsvariante in Form einer parametrierten Baumstruktur, auch Pruned Entity Structure (PES) genannt. In diesem Schritt wird ein Baum beschnitten, weshalb dieser Vorgang auch als *Prunen* oder *Prune*-Prozess bezeichnet wird. Gemäß den Abschnitten 2.1 und 3.2 sind alle Basisaufgaben, die die Blattknoten des Baumes darstellen, als ausführbare und parametrierbare Softwaremodule in einer Modellbasis (MB) abgespeichert. Im zweiten Schritt generiert der Steuerungsgenerator aus den Informationen der PES und den Komponenten der Modellbasis ein ausführbares Steuerungsprogramm gemäß dem in Abschnitt 2.2 dargestellten SBC Ansatzes und bringt dieses zur Ausführung. Zur Laufzeit der Steuerung ändern sich entsprechend dem realen Prozessverhalten die Attributwerte der Basisaufgaben fortwährend. Gemäß Abschnitt 3.3 kann dadurch ein Ereignis ausgelöst werden, welches eine neue Steuerungssynthese durch den Aufgabenplaner auslöst. Dieses Ereignis ist in Abbildung 8 mit „Prune“ bezeichnet. Wie im Schritt 0 in Abbildung 8 dargestellt, wird dazu dem Aufgabenplaner ein aktueller Parametervektor zur Analyse der Steuerungs_SES übergeben. Dieser Ablauf setzt sich iterativ fort, bis ein definiertes Abbruchkriterium eintritt.

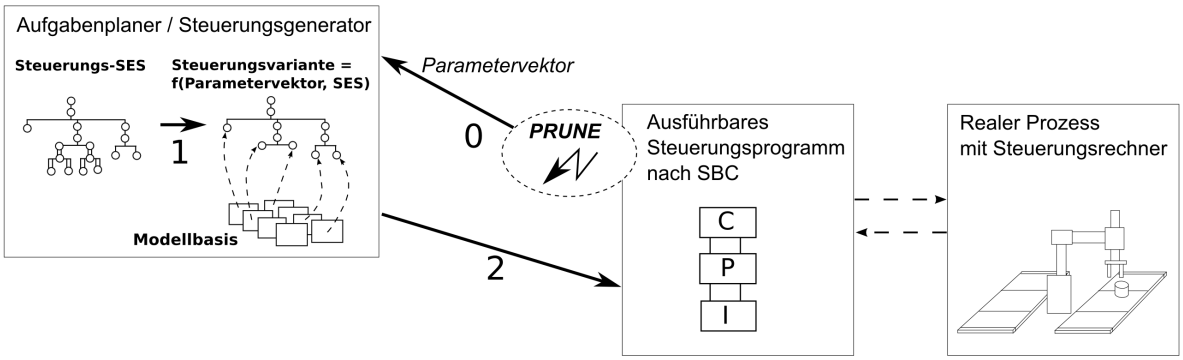

Abbildung 8: Automatische Generierung temporärer Steuerungsprogramme

An dieser Stelle wird die Modularisierung durch die FAR/SES Methode deutlich. Durch die strikte Trennung von Steuerungsbeschreibung (SES) und Softwaremodulen (Modellbasis) können gezielt einzelne Aufgabenmodule entwickelt und beliebigen Blattknoten der Steuerungs-SES zugewiesen werden. Weiterhin ist es problemlos möglich einzelne Module der Modellbais (MB) zu erweitern, da alle Wechselwirkungen innerhalb der Steuerungs-SES definiert sind und keine Wechselwirkungen unmittelbar zwischen den Modulen der MB existieren. Damit ist es auch möglich, die Entwicklung von Modulen der MB innerhalb eines Entwicklungsteams zu verteilen.

5 Erfahrungsbericht und Ausblick

Die aufgezeigte Methode (FAR/SES) zur systematischen Entwicklung flexibler aufgabenorientierter Robotersteuerungen auf Basis der System Entity Structure und einer Modellbasis (MB) ist prototypisch in der Entwicklungsumgebung MATLAB implementiert und getestet worden. Das Anwendungsbeispiel wurde vollständig mit der FAR/SES Methode

spezifiziert. Das iterative Prunen der Steuerungs-SES, zwecks Ableitung geeigneter Steuerungsvarianten, ist über eine Matlab-Schnittstelle mit SWI-Prolog realisiert worden. Die Softwaremodule der Modellbasis wurden dem DEVS-Formalismus [5, 6] folgend in Matlab implementiert und mittels eines DEVS Simulators ausgeführt. Dabei entsprach jedes DEVS-Modell einer Steuerungsvariante vollständig dem Simulationsmodellbasierten Steuerungsansatz. Aufgrund der Modularisierung mittels der FAR/SES Methode konnte die Modellbasis derart gestaltet werden, dass sämtliche Basisaufgaben der Robotermodule innerhalb der Steuerungs-SES miteinander austauschbar sind. Dieser Umstand erleichtert die reale Inbetriebnahme von Roboterapplikationen, da sich sämtliche Anpassungsarbeiten auf die Basisaufgaben der Modellbasis begrenzen. Die Schnittstellen zu den realen Prozeßkomponenten bildete die MatlabKK-Robotic Toolbox [8].
Zukünftige Arbeiten werden noch weitere Untersuchungen zur deklarativen Spezifikation von komplexen Steuerungen auf Basis der SES sein.

Literatur

[1] Haun, M.:
Handbuch Robotik.
Berlin, Heidelberg: Springer Verlag, 2007

[2] Maletzki, M.; Pawletta, T.; Pawletta, S.; Dünow, P.; Lampe, B.:
Simulationsmodellbasiertes Rapid Prototyping von komplexen Robotersteuerungen.
atp-Automatisierungstechnische Praxis,
Oldenbourg Verlag, München, 50(2008)8, 54 - 60

[3] Zeigler, B. P.; Hammonds, P.E.:
Modeling and Simulation-based Data Engineering
Burlington, San Diego, London: Elsevier Academic Press, 2007

[4] Abel, D.; Bollig, A.:
Rapid Control Prototyping, Methoden und Anwendungen.
Berlin, Heidelberg: Springer Verlag, 2007

[5] Zeigler, B. P. ; Prähofer, H. ; Kim, T. G. :
Theorie of Modeling and Simualtion, 2. Aufl.
San Diego, San Francisco, New York, Boston, London, ... : Academic Press 2000

[6] T. Schwatinski, T. Pawletta, S. Pawletta, C. Kaiser:
Simulation-based development and operation of controls on the basis of the DEVS formalism.
Proceedings of The 7th EUROSIM 2010 Congress,
Vol.2: Full Papers, Prag, Czech Republic, 2010, 8 pages

[7] Jacak, Witold:
Intelligent robotic systems: design, planning, and control.
New York: Kluwer Academic / Plenum Publishers, 1998

[8] Christern, Schmidt, Schwatinski, Pawletta:
KUKA-KAWASAKI-Robotic Toolbox for Matlab.
Hochschule Wismar,Website, 2011
http://www.mb.hs-wismar.de/cea/KK_Robotic_Tbx/KK_Robotic_Tbx.html