Global J. of Engng. Educ., Vol.10, No.3
Published in Australia

Optimierung des Engineeringprozesses durch die Integration von virtueller Realität

Thomas Zürcher
Höhere Fachschule für Technik Biel
Quellgasse 21, CH-2501 Biel, Schweiz

Die Bedürfnisse der Industrie sowie die überproportionale Zunahme der technologischen Komplexität verlangen eine Optimierung des herkömmlichen und weit verbreiteten Engineering-Verständnisses. Neue Trends und IT-Lösungen ermöglichen die Erschließung neuer Dimensionen. Damit diese in die industriellen Entwicklungsprozesse erfolgreich integriert werden können, müssen die Ingenieure bereits während ihres Studiums die Möglichkeit haben, mit diesen komplexen Strukturen und Zusammenhänge vertraut zu werden. Dies wiederum führt zur Stärkung der Wirtschaft.

DIGITALE FABRIK

Die weltumspannende Vernetzung und Internationalisierung der Industrie verlangt in zunehmendem Masse eine Flexibilität der Produktion. Der Anspruch an die Geschwindigkeit auch im Rahmen von Neuentwicklungen und die Verkürzung der *Time to Market* neuer Produkte prägen das Bild der modernen Fertigungsindustrie. Stillstand und zeitliche Verzögerungen werden in dieser hektischen Umgebung zu großen Bedrohungen und entscheiden oft über Erfolg und Misserfolg von Unternehmen. Es ist daher von zentraler Bedeutung, dass bei der Entwicklung und Weiterentwicklung von Produkten und Produktionsanlagen, alle relevanten Daten und Informationen, welche in irgendeiner Form mit dem Produkt oder der Anlage in Verbindung stehen, zentral gesammelt werden. Dieser zentrale Datenspeicher könnte somit als virtuelles Abbild des realen Produkts oder der realen Anlage angesehen werden. Je nach Bedarf kann aus dieser Sammlung von Daten eine virtuelle Anlage zusammengebaut werden, welche die gewünschten Parameter betont und der realen Anlage sehr nahe kommt. Durch ein gezieltes Einsetzen dieser virtuellen Anlagen kann der Entwicklungs- oder Änderungsprozess von Produkten und Anlagen optimiert werden [1].

Ziel dieser Publikation ist es, einen kleinen Aspekt dieser virtuellen Umgebung vorzustellen und anhand der umsetzungsorientierten Entwicklung den Bereich des prozessnahen simulationsbasierten Engineering aufzugreifen.

EVOLUTION IN DER AUTOMATION

Das Herzstück der modernen Fertigung sind hochautomatisierte Produktionsanlagen, welche durch die drei Hauptparameter Speed, Safety/Security und Reliability definiert sind. Durch das stark schwankende Umfeld des Marktes kommt zunehmend die Forderung nach möglichst flexiblen Produktionsmöglichkeiten auf. Die zunehmende Integration der modernen IT-Kommunikation ermöglicht einen hohen Informationsfluss, welcher zum einen eine Dynamisierung des Produktionsprozesses fordert und zum anderen diese Dynamisierung ermöglicht.

Daraus lassen sich gezielte Marktforderungen ableiten:

- Entwicklung neuer Anlagen gemäss den Konventionen verteilter Automatisierungssysteme, sowie der Integration bestehender Automationsanlagen. Dieser Schritt zur modularen Anlage erhöht die Beherrschbarkeit der zunehmend komplexer werdenden Technik;
- Die Durchgängigkeit der horizontalen wie vertikalen Kommunikation, einschließlich der Datenkommunikation, muss durch einen offenen Standard gewährleistet werden. Dazu gehören auch transparente Datenschnittstellen über alle Lebensphasen einer Anlagenautomatisierung von der Produktplanung inklusive CAD-Daten über die Anlagen Erstellung bis und mit Betrieb und Wartung;

- Die Produktionsdaten wie die Produktdaten müssen bis in die Unternehmensleitebene verfügbar sein und gegebenenfalls via Internet automatisch übermittelt werden können;
- Die Anforderungen an die Flexibilität von Produktionsanlagen gehen dahin, dass das Umstellen der Fertigung auf neue Produkte wesentlich schneller geht als bis anhin.

Durch die Summe dieser Forderungen wird deutlich, dass sowohl im Kommunikationsbereich wie im Engineeringbereich, Anpassungen und Weiterentwicklungen gemacht werden müssen. Diese Weiterentwicklungen werden auch in den Automatisierungstrends sichtbar.

Obwohl der Schritt von den zentralen Steuerungseinheiten hin zu den dezentral gesteuerten Automatisierungssystemen praktisch abgeschlossen ist, erkennt man den Trend zu Anlagen mit verteilten Steuerungshoheiten [2]. Dieser Trend wird zunehmend möglich, da die Rechenleistung der aktuellen Mikroprozessorsysteme zunimmt und die Preise der Elektronikkomponenten stetig sinken. Der Wandel von den zentralen zur verteilten Steuerungsintelligenz bringt den Nutzen, dass die Rechenleistung entsprechend den Anforderungen auf die ganze Anlage aufgeteilt werden können. Durch einen Umbau der Anlage wird dadurch die Rechenleistung automatisch den neuen Anforderungen angepasst. Durch diese dynamische Steuerungsanpassung erübrigt sich eine Überdimensionierung der Steuerung hinsichtlich späterer Anlagenerweiterungen. Ein weiterer markanter Vorteil dieser verteilten Projektierungsstrategie ist die konsequente Modularisierung von Großanlagen. Durch die Granularität, die bei der Projektierung der Anlage definiert werden muss, wird der Komplexitätsgrad der einzelnen Module bestimmt. Die Granularität einer Anlage hat somit einen direkten Einfluss auf die Wiederverwendbarkeit entwickelter Module. Diese Wiederverwertbarkeit ist somit ein interessanter Engineering-Aspekt, da bereits entwickelte Maschinenteile relativ einfach in neue Anlagenkonzepte integriert werden können.

In den weiteren Ausführungen wird die Entwicklung dieser Anlagenmodule analysiert und die Idee eines angepassten Engineeringprozesses verdeutlicht. Die Grundlage dieses neuen Engineeringansatzes stammt aus einer aktuellen Forschungsarbeit der ETH Zürich. Die Umsetzung im Bereich der komponentenbasierten Automationsentwicklung (CBA) ist ein aktuelles Forschungsgebiet der HFTbiel [3]. Dieses Projekt wird in enger Zusammenarbeit mit der Industrie (FA Kappeler und Glaus AG, Brügg) durchgeführt.

DER COMPONENT-BASED AUTOMATION (CBA) ANSATZ

Anlagen bestehen üblicherweise aus mehreren Teileinheiten, welche als technologische Module weitgehend autonom agieren und sich untereinander durch eine überschaubare Anzahl von Handshake-Signale koordinieren. Dem CBA – Ansatz liegen solche Module zugrunde und er richtet sich grundsätzlich nach der Programmiernorm IEC 61 499. In dieser wird die komponentenorientierte Programmierung definiert. Der CBA-Ansatz ist somit in weiten Teilen mit dieser Norm eng verbunden. Im Unterschied zur reinen Softwareentwicklung beinhalten automatisierte Anlagen nicht nur eine reine Softwarelösung, sondern sie bestehen im Wesentlichen aus den drei Einheiten: Mechanik, Elektrik/Elektronik und Logik/Software. Dies führt zu der in Abbildung 2 aufgeführten Überführung des technologischen Moduls in das Objektmodell einer Komponente.

Eine komplexe Anlage lässt sich somit als eine Verschaltung einzelner Komponenten darstellen, wobei jede Komponente die gesamte Information bezüglich Mechanik, Elektrik/Elektronik und Logik/Software beinhaltet [4].

Die Verschaltung solcher Komponenten muss über

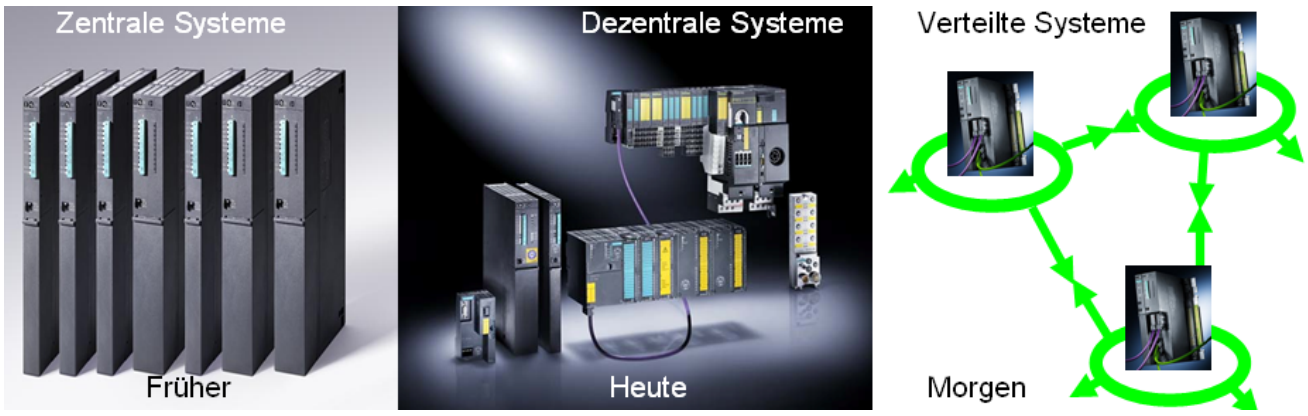

Abbildung 1: Evolution in der Automation.

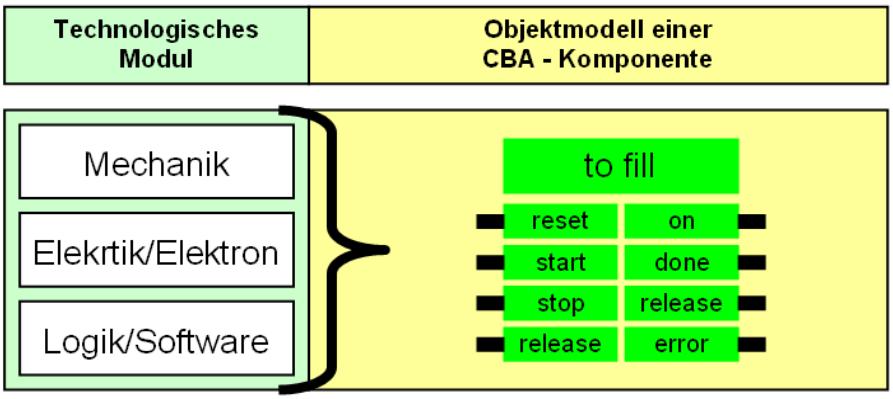

Abbildung 2: Komponentenmodellierung.

ein leistungsfähiges Kommunikationsnetz erfolgen, damit der Daten- und Steuerfluss gewährleistet werden kann. In diesem Bereich bieten sich Lösungen von Industrial Ethernet an. Die hohe Kommunikationsleistung bildet die Basis dieses intermodularen Datenverkehrs.

Im Zentrum der aktuellen Forschungsarbeit der HFTbiel steht nicht direkt der *Component-Based Automation* Ansatz, sondern die daraus resultierende modulare Struktur komplexer Automationsanlagen. Dieses in technologische Module aufgeteilte Anlagendesign bietet sich für eine Überarbeitung des traditionellen Entwicklungszyklus für elektromechanische Anlagen an. Die daraus resultierenden Optimierungen werden als signifikant erachtet und beeinflussen den Lifecycle einer Anlage nachhaltig positiv.

Um die Idee des optimierten Engineeringprozesses zu verdeutlichen, wird der Focus auf die Realisierung eines technologischen Moduls gerichtet. Dieses Modul gilt synonym für alle gekapselten mechatronischen Prozesse mit einer begrenzten Komplexität.

REALISIERUNG EINES TECHNOLOGISCHEN MODULS

Der eigentliche Knackpunkt zu Beginn der Engineeringphase liegt darin, die Granularität einer komplexen Anlage zu definieren. Wird diese zu klein gewählt, steigt der Komplexitätsgrad der zu realisierenden Anlage stark an. Wird sie aber zu grobkörnig ausgelegt, ist es schwierig die technologischen Module nach der Realisierung für weitere Projekte zu verwenden. Diese Modular-isierung kann daher nur von einem interdisziplinären Ingenieurteam in gemeinsamer Diskussion vorgenommen werden. Der Grund dafür liegt in der Definition des technologischen Moduls [5].

Grob gesehen kann ein technologisches Modul in zwei Hauptbereiche eingeteilt werden. Den Physical-Body, welcher alle physikalischen Eigenschaften eines Moduls beschreibt. Wichtig zu erkennen ist hier, dass der Physical-Body nicht ausschließlich von der mechanischen Konstruktion definiert wird, sondern die physikalischen Eigenschaften der Antriebe und des elektronischen Zubehörs miteinbezogen werden müssen.

Entsprechend werden dem Logical-Body die Software, die Steuerungskomponenten und die logischen Eigenschaften der Antriebe sowie des elektronischen Zubehörs zugeordnet.

Betrachtet man den traditionellen Engineeringprozess, stellt man fest, dass in den meisten Fällen in der Konzeptphase mit der mechanische Definition einer Anlage begonnen wird. Während der Entwurfsphase definiert der Konstrukteur das definitive

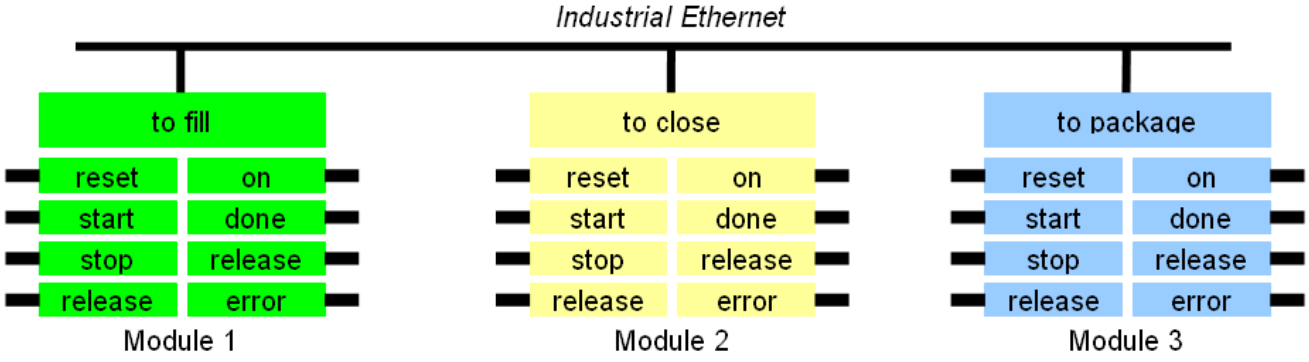

Abbildung 3: Komponentenverschaltung mittels *Industrial Ethernet*.

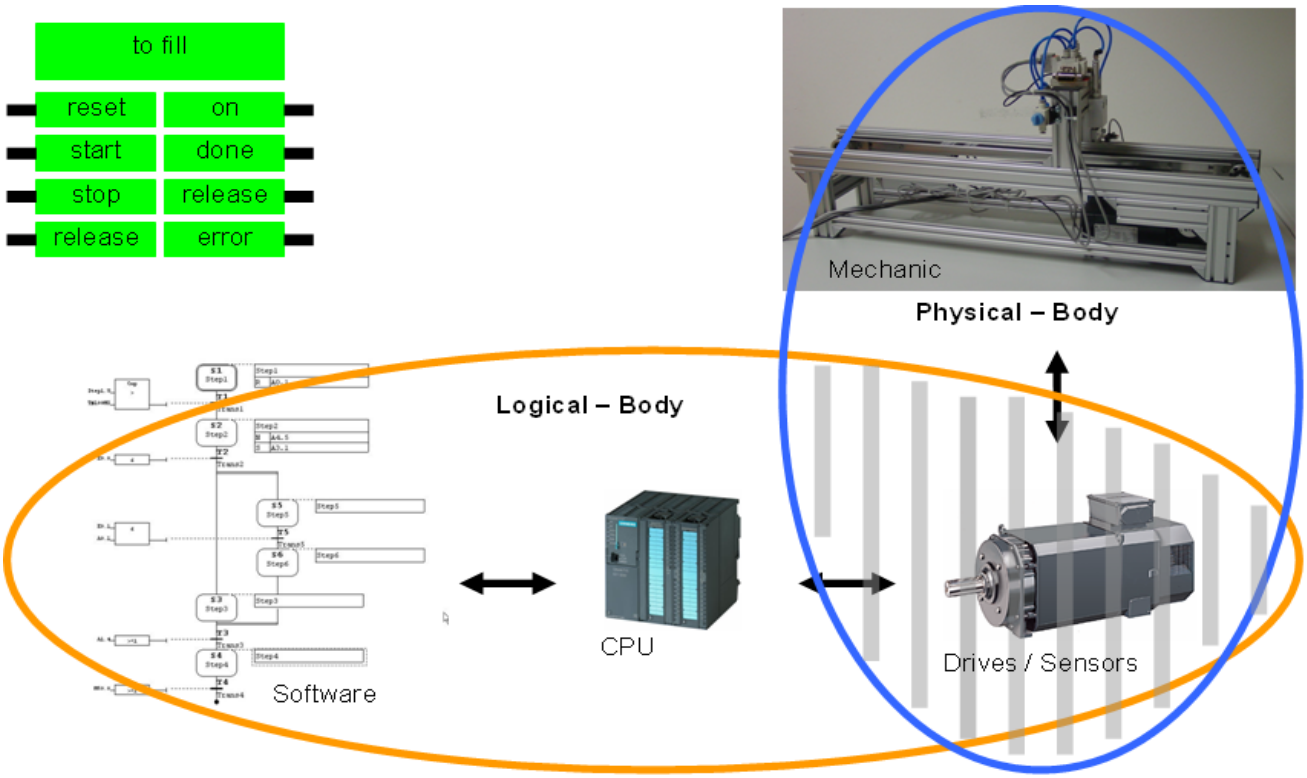

Abbildung 4: Aufbau des technologischen Moduls.

mechanische Anlagendesign und beginnt dann in Zusammenarbeit mit dem Elektroingenieur die Anlage mit der Auswahl und Integration elektrischer und elektronischer Komponenten zu komplettieren. Durch diese späte Kontaktaufnahme ist es meist nicht mehr möglich ein optimales Design der mechatronischen

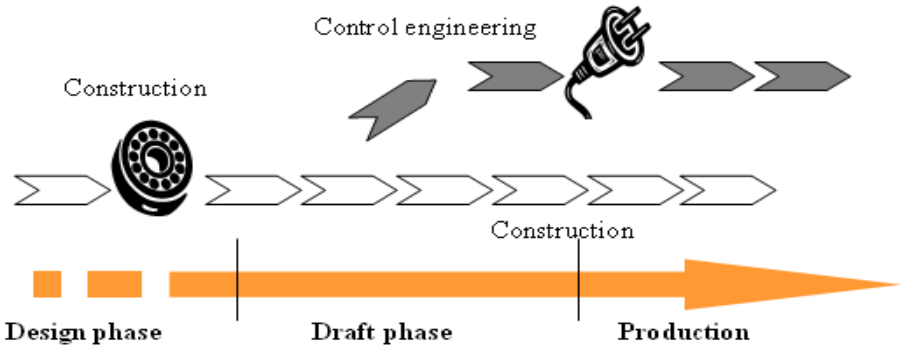

Abbildung 5: Traditioneller Engineeringprozess.

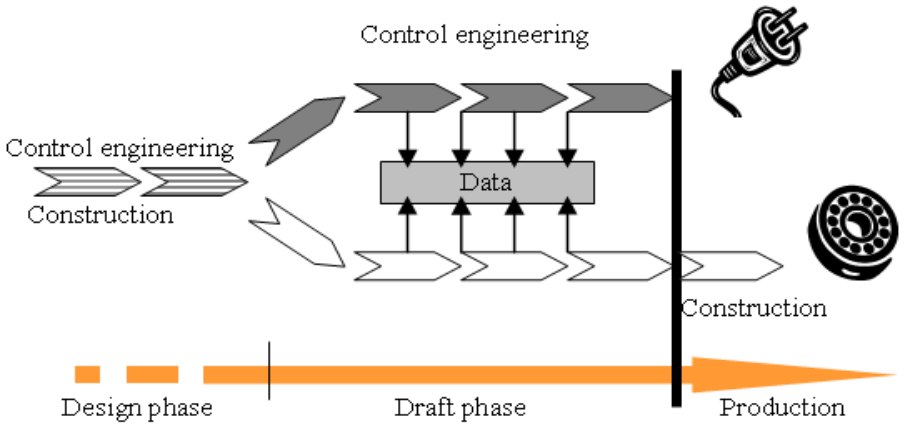

Abbildung 6: Optimierter Engineeringprozess Grundlage ETHZ.

Komponenten zu erzielen, da in der Regel die Konstruktion schon zu weit fortgeschritten ist. Ein weiterer Nachteil ist, dass erst mit der Realisierung der Anlage die eigentliche Softwareentwicklung der Steuerung in Angriff genommen werden kann. Es ist bekannt, dass der Hauptentwicklungsteil der Steuerung stets in der Projektschlussphase sehr intensiv wird, sich hinzieht und oft zu massiven Verzögerungen der Fertigstellung führt. Komplikationen, die in dieser Projektphase erkannt werden, führen zu kostenaufwändigen Änderungsprozessen.

Bringt man nun den aktuell an der ETH Zürich diskutierten Engineeringprozess mit dem aus dem CBA-Gedanken resultierenden technologischen Modul zusammen, so erkennt man, dass der Engineeringprozess wesentlich verkürzt werden kann.

Durch die gemeinsame Konzeptphase wird erreicht, dass bereits vor der Entwurfphase eine in sich stimmige elektromechanische Grundlage definiert wird. Die während der Entwurfphase ausgearbeiteten CAD-Zeichnungen, Elektronikfunktionen, Drivespezifikationen, usw. werden in einer zentralen Datenbank gesammelt. Diese Datenbank beinhaltet alle Informationen, des zu realisierenden technologischen Moduls. Dieses ist somit virtuell komplett definiert. Das heißt der vollständige 3D-CAD Konstruktionssatz definiert das virtuelle Erscheinungsbild des technologischen Moduls und die Drivespezifikationen sowie elektronischen Funktionen definieren den funktionellen Teil. Das bedeutet, dass das technologische Modul virtuell vollständig zusammengesetzt werden kann.

Gelingt es nun diese Daten zu einem virtuellen Prozess zusammen zu fügen und diesen mit einer echten Steuerung zu koppeln, so wäre der Softwareentwicklungsingenieur nicht mehr auf das fertig gestellte Modul angewiesen. Die Steuerungssoftware kann somit erstellt und am virtuellen Prozess getestet werden. Komplikationen, welche erst bei der Inbetriebnahmephase eines technologischen Moduls erkannt werden, können somit unter Umständen zu einem früheren Zeitpunkt entdeckt und behoben werden. Der Hauptvorteil liegt in der Tatsache, dass die Inbetriebnahme der eigentlichen Anlage mit einer bereits getesteten und optimierten Software geschieht. Beliebig komplexe Anlagen können somit durch eine Vielzahl simulierter technologischer Module problemlos zusammengesetzt und über die echte Steuerung miteinander vernetzt werden.

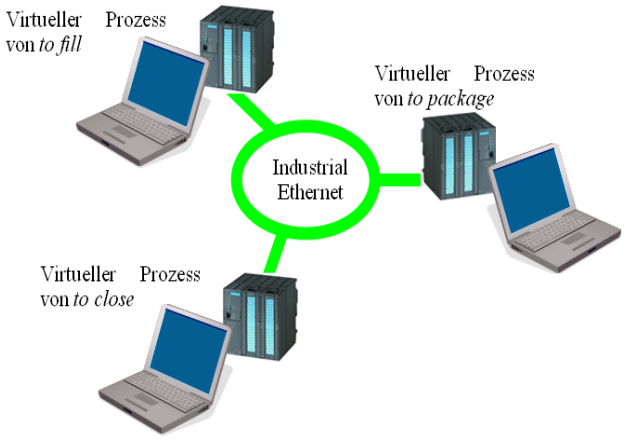

Abbildung 7: Simulation komplexer Anlagen.

EIN SIMULATIONSANSATZ MIT OFFENEN SCHNITTSTELLEN

Mit dem Simulationswerkzeug *ViSiWorks* der Firma Kappeler und Glaus AG, Brügg, Schweiz wird die Virtualisierung eines technologischen Moduls ermöglicht.

ViSiWorks ist ein Simulationswerkzeug, welches intern mit einem dreidimensionalen Abbild sämtlicher Objekte arbeitet und so eine virtuelle Realität erzeugt.

Alle Objekte werden einzeln mit ihren physikalischen Eigenschaften wie Position im Raum, Länge, Breite und Geschwindigkeit verwaltet. Dies, verbunden mit sehr kurzen Zykluszeiten für die Datenübermittlung zu den Steuerungen, ermöglicht ein äußerst reales Verhalten der simulierten Anlage.

ViSiWorks ist voll konfigurierbar. Anlagen werden aus einzelnen vordefinierten Komponenten zusammengestellt. Der Aufbau und die Bedienung der virtuellen Anlage gestalten sich wie der Aufbau einer Modelleisenbahn.

Der kurze Softwareüberblick soll einen kleinen Einblick in die optimierte Simulationsstruktur geben. Wichtig bei dieser Art von Simulation ist die Simulationsgeschwindigkeit, da im Steuerungsbau mit Real-time-Applikationen gearbeitet wird. *ViSiWorks* arbeitet diesbezüglich sehr effizient. In ersten Tests wurden bereits komplexe Förderanlagen mit Erfolg modelliert und simuliert [6].

AUFBAU DER SIMULATION

In einer 3D-Simulation sind diverse Körper gegeben. In *ViSiWorks* können für diverse Körper resp. Objekte (Graphik, Kollisionskörper, Funktionskörper) Dateien im VRML-Format angegeben werden.

Die eventuell aus dem Konstruktionsprozess vorhandenen 3D-CAD-Anlagedaten könnten nun in das VRML Format umgewandelt resp. exportiert werden. Bei Kollisions- und Funktionskörper ist die Einschränkung zu machen, dass *ViSiWorks* nicht genau den im File angegebenen Körper erzeugt, sondern eine Box mit den Maximalabmessungen des Körpers.

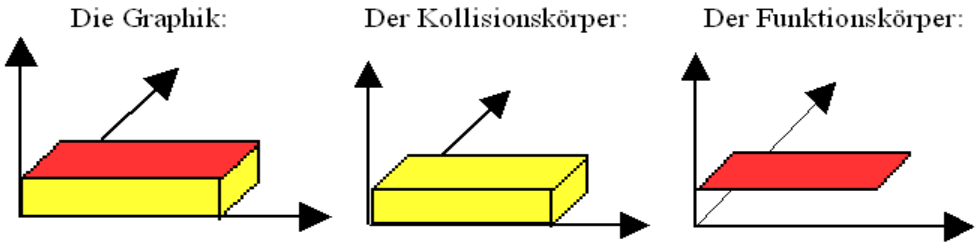

Abbildung 8: Aufbau von Simulationsobjekten in ViSiWorks.

Aufbau und Konzept von Komponenten in *ViSiWorks*

Eine *ViSiWorks*-Komponente besteht aus den vier Teilen: System-Komponente, Graphik-Klassen, Physik-Klassen und Funktions-Klassen.

Jede Klasse hat ein Präfix V für *ViSiWorks* und DV für Data-*ViSiWorks*. So verfügt jede V-Klasse über eine entsprechende DV-Klasse, in welcher die Daten enthalten sind. Beim Speichern und Laden einer Komponente werden nur die DV-Klassen berücksichtigt.

Man kann vereinfacht sagen, dass V-Klassen nur Code (Logik) enthalten (natürlich abgesehen von Hilfsattributen) und DV-Klassen nur Attribute (Daten).

Die Beschreibung des Verhaltens und des Aussehens der Komponente befindet sich in den Klassen VGraphical, VPhysical und VComponentFunction, die Klasse VSystemComponent dient nur als Zugangspunkt zur Komponente und zur Verwaltung der drei Zeiger.

Die Klasse VSystemComponent

Die System-Komponente entspricht der eigentlichen Komponente. Der Hauptzweck dieser Klasse ist es, die Zeiger auf die Graphik-, die Physik- und die Funktionsklasse zu definieren. Sie behandelt das Informieren der drei Klassen, falls sich in den Daten etwas ändert.

Die Klasse DVDescription

In der DVDescription-Klasse sind die gewünschten Klassennamen für die Graphik-, Physik- und

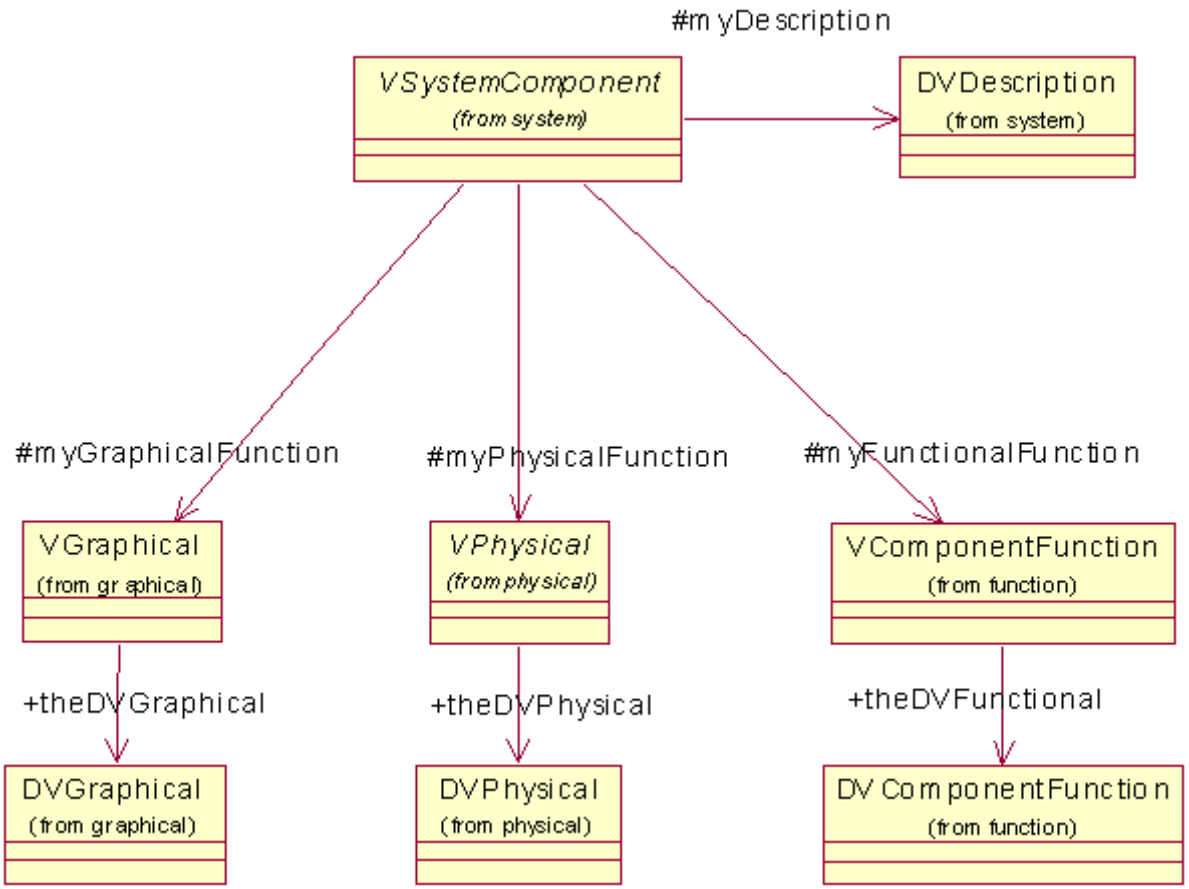

Abbildung 9: Softwaremodell von ViSiWorks.

Funktionsklassen angeben. Die Klassennamen müssen mit allen Packagenamen angegeben werden.

Graphik

Text hat keine direkte Verbindung zur Steuerung und reagiert nicht auf Aktoren und beeinflusst keine Sensoren. Die Klasse VGraphical ist für das Erzeugen der Graphik am Bildschirm verantwortlich. Die dargestellte Graphik muss nicht identisch sein mit der Funktionsfläche oder dem Kollisionskörper.

In der Klasse DVGraphical werden die Zeiger auf das entsprechende Graphik-File angegeben. Außerdem gibt es noch diverse Formatierungsoptionen, z.B. die Skalierung der Graphik oder, falls ohne Graphikfile gearbeitet wird, die Farbe der Komponente.

Die Graphikfiles müssen im 3D-Format VRML vorliegen. Unterstützt wird nur die Version 2.0, auch VRML97 genannt.

Wie oben erwähnt, muss bei einfachen Komponenten nicht unbedingt ein Graphikfile vorhanden sein, die entsprechenden Pfad-Attribute können *null* sein. In diesem Fall wird eine Box mit den Dimensionen, wie in der Klasse DVPhysical angegeben, erzeugt.

Physik

Zum Beispiel: Gebinde, Nocken und Stopper haben keine direkte Verbindung zur Steuerung, sie reagieren jedoch auf einzelne Aktoren und beeinflussen einzelne Sensoren.

Die Klasse VPhysical ist für das physikalische Verhalten der Komponente verantwortlich. Sie bestimmt z.B. mit welchem Körper die Komponente mit anderen Komponenten kollidieren kann, oder ob die Komponente der Schwerkraft unterworfen ist.

Komponenten, die nicht zur Kategorie VPhysicalRelevantComponent gehören, dürfen nicht über eine VPhysical- Klasse verfügen.

Die Klasse DVPhysical enthält alle Angaben zur Physik der Komponente. Dies umfasst alle Informationen zur Position im Raum und zum Kollisionskörper.

Das File des Kollisionskörpers muss im 3D-Format VRML vorliegen. Unterstützt wird nur die Version 2.0, auch VRML97 genannt.

Wie bei der Graphik auch, muss bei einfachen Komponenten nicht unbedingt ein File für den Kollisionskörper vorhanden sein, das entsprechende Pfad-Attribut kann *null* sein. In diesem Fall wird eine Box mit den in den Attributen *length, depth, height* angegebenen Dimensionen erzeugt.

Ist ein File angegeben, werden *length, depth, height* aus dem File-Körper berechnet.

Funktion

Sensoren (Lichtschranken, Taster und Nockenschalter setzen Eingänge der Steuerung) Aktoren (Ketten- und Rollenförderer(respektive deren Antriebe)) sowie Lampen reagieren auf die Ausgänge, welche von der Steuerung gesetzt werden.

In der Klasse VComponentFunction liegt der größte Einflussbereich des Benutzers. Hier kann er bestimmen, wie sich eine Komponente verhalten soll, wie sie z.B. auf einen Mausklick oder auf den Zustand eines Ausganges reagiert.

Dazu kann der Benutzer je nach Bedarf diverse Listen überschreiben und dort mit der Funktion von Mikrokomponenten verknüpfen.

Falls die Komponente auf einen Mausklick reagieren soll, wird vorgeschlagen, dass dies ebenfalls hier geschieht. Dazu wird eine Funktion definiert, welche in der entsprechenden Methode des MouseListener, der sich in der Klasse VSystem-Component befindet, aufgerufen wird.

Die Klasse DVComponentFunction enthält die Angaben zur Funktion der Komponente. Dies umfasst Informationen zu eventuell vorhandenen Mikrokomponenten und zur Position des Funktionskörpers im Raum. Als Funktionskörper wird der Bereich verstanden, indem die Funktion der Komponente mit anderen Komponenten interagiert. Dies ist bei einer Lichtschranke z.B. der Schaltbereich, oder bei einem Förderer die Transportfläche.

Das File des Funktionskörpers muss im 3D-Format VRML vorliegen. Unterstützt wird nur die Version 2.0, auch VRML97 genannt.

Wie bei der Graphik auch, muss bei einfachen Komponenten nicht unbedingt ein File für den Funktionskörper vorhanden sein, das entsprechende Pfad-Attribut kann *null* sein. In diesem Fall wird eine Box mit der in den Attributen *length, depth, height* angegebenen Dimensionen erzeugt. Mit den Werten x1, y1, z1 wird angegeben, um welche Distanz der Funktionskörper zum Nullpunkt der Komponente versetzt ist.

ZUSAMMENFASSUNG

Das Simulieren komplexer mechatronischer Anlagen als Ganzes, führt zu Problemen mit der Rechenleistung. Dies hat massive Einflüsse auf die Simulationsgeschwindigkeit. Damit die Simulation jedoch erfolgreich zur Softwareentwicklung eingesetzt werden kann, muss eine Mindestgenauigkeit im Real-time-Kommunikationsbereich gewährleistet werden. Dies wird erreicht indem der CBA-Gedanken

kompromisslos zum Einsatz kommt. Das heißt die Komplexität der gesamten Anlage wird auf die Simulation einzelner technologischer Module aufgeteilt und somit ist die benötigte hohe Simulationsgeschwindigkeit gewährleistet. Dies ermöglicht die Simulation, welche die Grundlage des optimierten Engineeringprozesses bildet. Dieses Vorgehen ermöglicht massive Zeit- und Kostenersparnisse in der Entwicklung von Anlagen.

Damit dieser Gedanke in der industriellen Umsetzung bestmöglichst Fuß fassen kann, ist es unerlässlich, dass die Ingenieure bereits während ihres Studiums mit diesem neuen Engineeringprozess vertraut werden.

REFERENZEN

1. Pigan, R. und Metter, M., Automatisieren mit PROFINET. Erlangen: Publics Corporate Publishing (2005).
2. Zürcher, T., Fernprogrammierung und Ferninbetriebnahme Mechatronischer Anlagen. *Proc. 4th Inter. Symp. on Automatic Control*, Wismar, Deutschland (2005).
3. Felser, M., PROFINET CBA Systembeschreibung Ausgabe 2004. Burgdorf: BFH (2004).
4. Zürcher, T., Remote sensing and remote commissioning. *Proc. 6th Inter. Workshops on Research and Educ. in Mechatronics*, Annecy, Frankreich (2005).
5. Heinzelmann, E., Technische Rundschau. August (2005).
6. Zürcher, T., Simulation-based engineering for creating and implementing mechatronic processes. *Proc. Jubilee Inter. XV Symp. of Micromachines and Servosystems*, Soplicovo, Polen (2005).

BIOGRPHIE

Thomas Zürcher ist stellvertretender Direktor der Höheren Fachschule für Technik Biel und leitet den Fachbereich Automation. Als Dozent der Automatisierungstechnik unterrichtet er die Module *Steuerungs- und Kommunikationssysteme integrieren* sowie *Projektorientiertes Engineering*.
Er war unter anderem als Projektingenieur im internationalen Wasserkraftwerkbau tätig, während dieser Zeit betreute er den Bereich Steuerungs- und Kommunikationstechnik. Er entwickelte Konzepte und Lösungen für die Fernsteuerung, Fernüberwachung und Fernalarmierung von Wasserkraftanlagen.