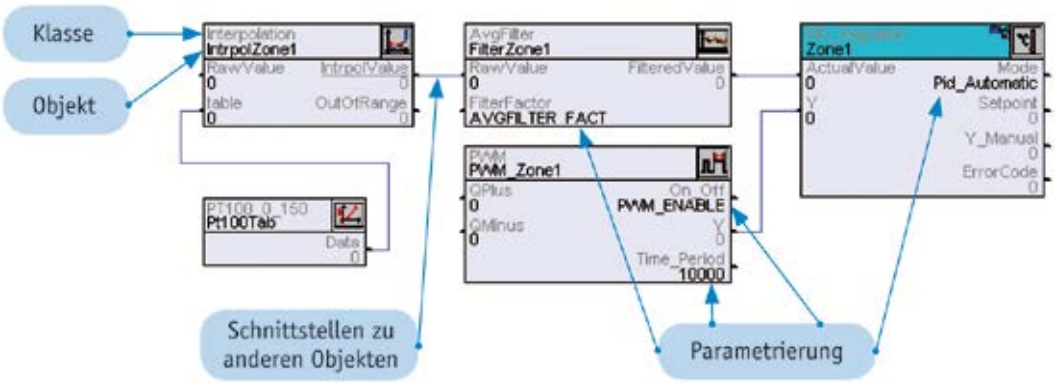
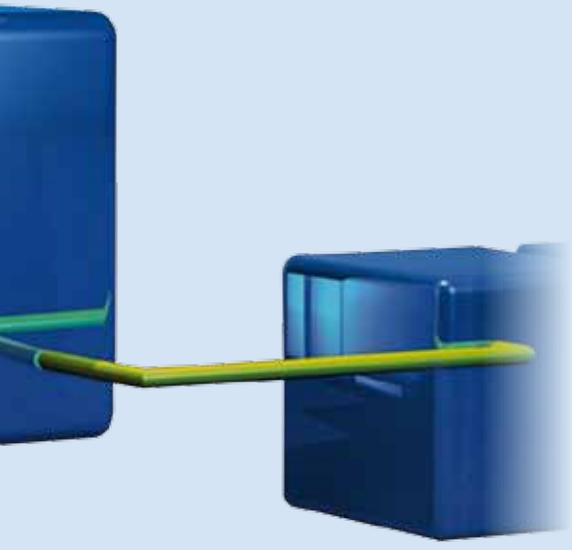
Funktionsarchitektur mit Klasse

Die Komplexität im Maschinen- und Anlagenbau zeigt keine abnehmende Tendenz, ebenso wenig der Preisdruck. Zugleich kaufen die Kunden die Katze nicht gern im Sack, sondern verlangen mit dem Angebot den Nachweis, das zu bekommen, was sie bestellen werden. Diesen Spagat kann nur schlagen, wer sich von traditionellen, sequenziellen Methoden in der Softwareentwicklung verabschiedet und statt wild drauflos zu programmieren Funktionsarchitektur betreibt. Das braucht eine durchgängige Entwicklungsumgebung, die diese Vorgehensweisen über alle Unterdisziplinen hinweg konsequent unterstützt. Wie LASAL von Sigmatek.

Autor: Ing. Peter Kemptner / x-technik



Objektorientiertes Programmieren erleichtert das Handling komplexer Programme, da diese in handhabbare Teilkonstrukte gegliedert werden. Die sogenannten Klassen werden durch individuelle Parametrierung zu Objekten, die als Programmbausteine eigenständig sind und beliebig zusammengestellt werden können.



„Das Rollenbild des Automatisierungs-Softwareentwicklers hat sich gewandelt“, sagt Franz Aschl, Innovationsmanagement bei Sigmatek. „Waren sie früher mit der Übersetzung von Ablauf-Anforderungen in Programmierungen beschäftigt, ermöglichen ihnen heute objektorientierte Programmierung (OOP) und integrierte Entwicklungsumgebungen mit hoher Usability wie LASAL, als Funktionsarchitekten die Entstehung mechatronischer Produkte bestimmend zu gestalten.“

Um diese Aussage in seiner gesamten Tragweite zu verstehen, ist vielleicht ein geschichtlicher Rückblick hilfreich: Der Maschinenbau hat eine Jahrtausende lange Geschichte. Schon im klassischen Altertum wurden recht komplexe Maschinen hergestellt. Ob Flaschenzüge und Rollenbatterien zur Erleichterung von Transportaufgaben, Sprengmechanismen zur Gewinnung von Baustoffen oder Katapulte als Waffensysteme. Sie alle hatten eine Gemeinsamkeit: Diese Vorrichtungen zur Kraftübertragung waren rein mechanisch aufgebaut und erfüllten eine exakt umrissene Funktion.

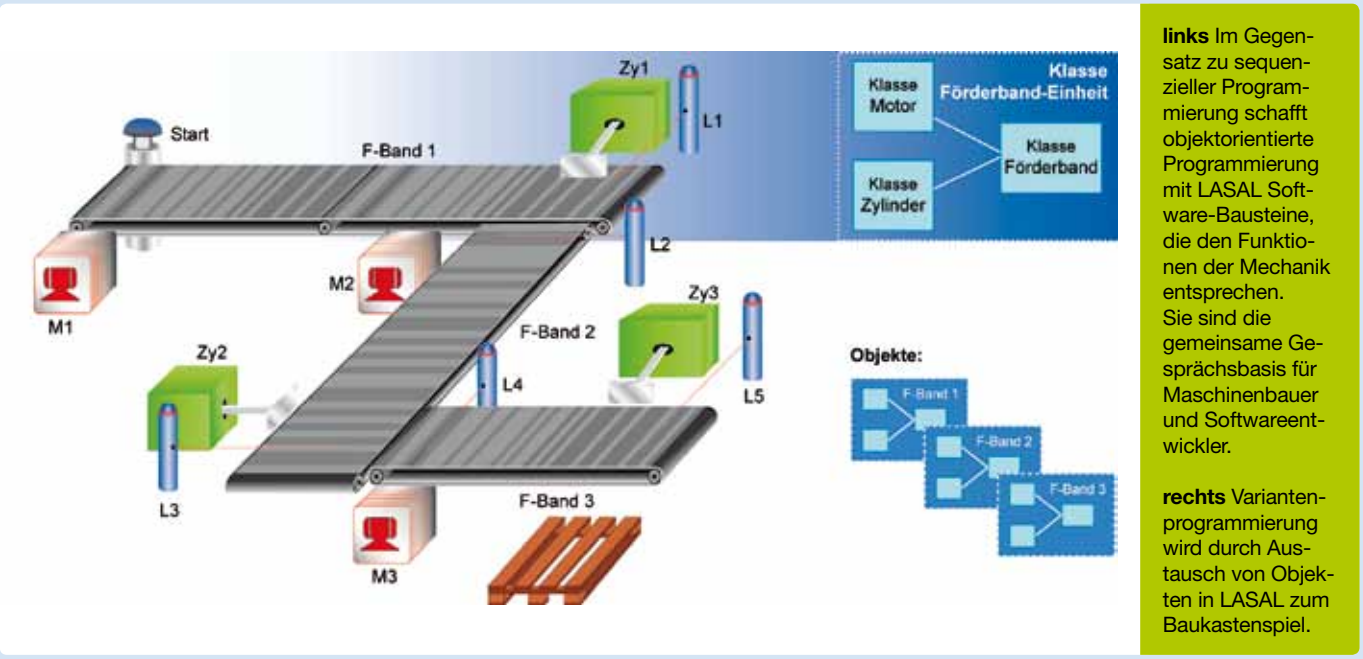
Von reiner Funktion zur Automation

Zu Automaten werden Maschinen durch die zusätzliche Dimension des zeitlichen Ablaufs. Dessen Steuerung beschäftigt die Ingenieure ebenfalls schon sehr lang. Viele Technologien wurden dazu angewendet, doch erst die Elektronik machte die Steuerung auch komplexerer Abläufe nach logischen Mustern möglich. Den Durchbruch brachte die speicherprogrammierbare Steuerung, die deshalb auch mehrere Jahrzehnte lang in der Industrieautomatisierung den Ton angab.
Erstmals 1968 bei GM erdacht, sollten SPS schnell und kostengünstig zu entwickeln, die Programmierung leicht zu modifizieren sein. Im Vergleich zur zuvor eingesetzten Technik mit Relais oder hart verdrahteter Logik war sie das auch. Was sich nicht änderte, war die Denkweise der Automatisierer. Sie orientierte sich an der Logik-Verschaltung bei der Programmierung mittels Kontaktplan (KOP) oder an der sequenziellen Arbeitsweise der Steuerrechner bei Programmierung per Anweisungsliste (AWL). Auch die Verwendung von Hochsprachen wie C änderte nichts an dieser eindimensional ablauforientierten Methodik. Diese ist eigentlich unnatürlich, weil sie kein direktes Abbild der Funktion der Maschine oder des Maschinenteils darstellt. Sie macht eine Übersetzung erforderlich zwischen der Logik der mechanischen Zusammenhänge und der Logik der Befehlsabarbeitung in der Steuerungselektronik.

Immer leistungsfähigere Hardware und immer umfangreichere Funktionalitätsanforderungen an den Maschinenbau führen dazu, dass solche Programme immer größer werden. Selbst bei diszipliniert strukturiertem Aufbau ist es wegen ihrer inneren Verwobenheit sehr schwierig, sie nachträglich zu ändern oder für eine partielle Wiederverwendung zu teilen.

Mit Objekten zurück zur Funktion

Bereits vor etwa 20 Jahren erfolgte daher in manchen Branchen der Softwareentwicklung eine Abkehr von strikt sequenzieller Programmierung. Sie wurde ersetzt durch den objektorientierten Ansatz, bei dem jede einzelne Funktion in einem als Klasse bezeichneten handlichen Block gekapselt ist, sodass eine ungewollte Beeinflussung von außen ausgeschlossen werden kann. In ihrem Inneren enthalten diese Klassen die bekannte Programmierung per Structured Text (ST), Anweisungsliste (AWL), →



links Im Gegensatz zu sequenzieller Programmierung schafft objektorientierte Programmierung mit LASAL Software-Bausteine, die den Funktionen der Mechanik entsprechen. Sie sind die gemeinsame Gesprächsbasis für Maschinenbauer und Softwareentwickler.

rechts Variantenprogrammierung wird durch Austausch von Objekten in LASAL zum Baukastenspiel.

Kontaktplan (KOP) nach IEC 61131-3 oder in ANSI-C.

Mit Parametern und Schnittstellen versehen, bezeichnet man diese Programmbausteine als Objekte. Sie lassen sich beliebig zu ganzen Programmen zusammenstellen. Der Vorteil dieser Entwicklungsmethode ist ihre durchgängige Modularität von der untersten Ebene der einzelnen Funktion bis hinauf zum Gesamtprojekt, die durch hierarchische Gruppierung und Verbindung von Programmobjekten wie in einem Baukastensystem entsteht. Das sorgt für Übersichtlichkeit und zwingt zu strukturierter Software-Entwicklung. Der gefürchtete „Spaghetti-Code“ kann gar nicht entstehen. Und Funktionen können einzeln oder als Gruppen entwickelt, getestet, hinzugefügt, ausgetauscht oder bei Nicht-Verwendung einer Option ausgeblendet werden.

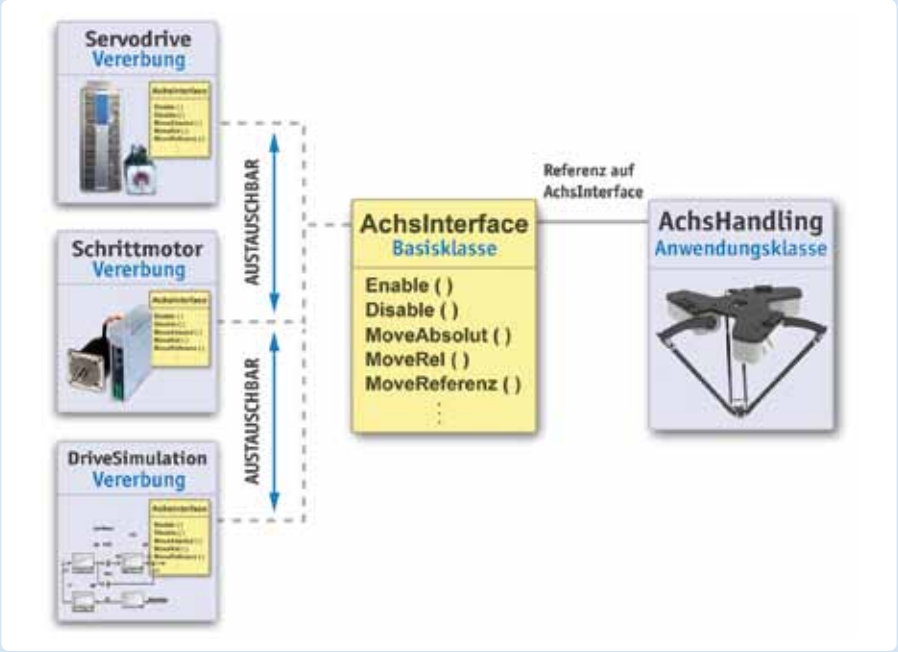
Durchgängig entwickeln: vom Konzept zum Detail

„Ein wesentlicher Vorteil der objektorientierten Programmierung ist, dass sie zu wohlüberlegter Formulierung der Definitionen zwingt“, findet DI (FH) Bernd Hildebrandt, Vertriebsleiter Österreich bei Sigmatek. „So erfolgt zuerst die Festlegung der funktionalen Architektur. Erst wenn diese das OK erhalten hat, wird Arbeit in die Detailprogrammierung gesteckt.“ Dass diese einen geringeren Aufwand als bei traditioneller Programmierweise darstellt, liegt nicht nur an den Möglichkeiten zur Wiederverwendung und Eigenschaftsvererbung der Klassen oder an den umfangreichen Bibliotheken mit vorgefertigten Klassen in LASAL. Das größte Potenzial zur Einsparung von Programmierzeiten liegt darin, dass durch die frühzeitige Überprüfbarkeit nur einmal programmiert werden muss.

Diese Möglichkeiten von LASAL versetzen Projektleiter in die Lage, bereits in der Projektierungsphase die gesamte Automatisierungslösung quasi wie ein Architektenmodell im Maßstab 1:1 zu entwerfen, die Machbarkeit zu überprüfen und den Entwicklungsaufwand abzuschätzen, auch wenn noch nicht alle später benötigten Programmteile existieren. So kann die weitere Entwicklung auf Basis eines funktional abgestimmten Programmgerüstes erfolgen, das auch bereits die meisten Schnittstellendefinitionen enthält. „Der Clou ist, dass Softwareentwickler und Maschinenbauer wieder in den gleichen Begriffen denken“, sagt Franz Aschl. „Sie finden über diesen funktionsorientierten Entwicklungsansatz eine gemeinsame Sprache, sodass die Missverständnisse zwischen ihnen weniger werden.“

„Die objektorientierte Programmierung würde jedoch zu kurz greifen und sich ad absurdum führen, bliebe sie auf die Ablaufsteu-

Wie die Steuerung und die Antriebstechnik erfolgt auch die Visualisierung in LASAL auf Basis der Objektorientierung. So kann auch die Ergonomie eines Bildschirminhaltes bereits frühzeitig überprüft werden, auch wenn manche Objekte noch nicht den endgültigen Inhalt haben. Hier das Beispiel eines Service-Menüs mit Steuerungskomponenten in Schaltschrankansicht.



erung begrenzt“, weiß Bernd Hildebrandt. „Die Integration der Automatisierung ist nur total, wenn auch die Antriebstechnik, die Visualisierung, die Sicherheitstechnik und Numerik als Teilfunktionen der Gesamtmaschine mit derselben einheitlichen Methodik zu realisieren sind.“ So kann etwa die frühzeitige Definition der Benutzeroberfläche nicht nur der Abstimmung der Entwicklungsziele mit dem Kunden dienen, sondern auch der Festlegung nachgelagerter Funktionen, die daher auch keine doppelte Dateneingabe brauchen.

Zu den Komfortfunktionen von LASAL gehört, dass noch nicht mit Inhalten versehene Systemteile innerhalb der Baumstruktur aus vorhandenen und definierten, aber vorerst leeren Klassen mit LARS, einem Windows-basierten Simulationstool, simuliert werden können. So muss mit dem Testen fertig gestellter Softwarekomponenten nicht auf das Vorhandensein sämtlicher Teile gewartet werden, was die Entwicklungszeit weiter verkürzt.

Wissenserhalt und Komfort

„Heutige HTL- oder FH-Abgänger sind bereits mit objektorientierten Software-Entwicklungsmethoden aufgewachsen“, stellt Franz Aschl fest. „Sie sind es gewohnt, modular und funktionsorientiert zu denken und ihre Entwürfe zu überprüfen, bevor sie zu codieren beginnen.“ Sie erinnern sich nicht an Zeiten, als Programme auf wenigen Kilobytes Platz finden mussten und daher von selbst entsprechend überschaubar waren. Sie denken in Vielfachen von Megabytes und haben wenig Respekt vor der Komplexität, die zu beherrschen sie als ihre Aufgabe ansehen.

Softwareentwickler müssen in begrenzter Zeit Automatisierungslösungen mit nachvollziehbar hoher Softwarequalität schaffen, die sich zudem einfach wiederverwenden lassen. Dazu brauchen sie eine durchgängige Entwicklungsumgebung, innerhalb derer sie über sämtliche Projektphasen hinweg die Funktionalität ihrer Produkte gestalten können. Nicht nur von der Pflichtenhefterstellung über die Detailprogrammierung bis zur Inbetriebnahme, sondern über den gesamten Produktlebenszyklus für Softwarewartung und Erweiterungen.
Gefragt ist ein Engineering Tool mit intuitiver Bedienung, komfortablen Möglichkeiten zur Simulation und Visualisierung aller Prozesse, mit einer vollständigen Testumgebung bis hin zum Debugger und mit zentraler Verwaltung von Projekten und Versionen. So kann die Gesamtaufgabe mit geringem Abstimmungsaufwand auf mehrere Schultern verteilt werden. „Mich wundert wirklich, dass andere, auch sehr namhafte Automatisierungs-Systemhersteller erst jetzt beginnen solche Systeme vorzustellen, und das zum Teil sehr zögerlich“, sagt Bernd Hildebrandt. „Von Sigmatek gibt es diese Softwareumgebung mit Klasse für alle Projektphasen bereits seit zehn Jahren. Sie heißt LASAL.“

Die Vorteile von LASAL erkannt hat zum Beispiel VIET Italia s.r.l., ein Hersteller von Schleif- und Poliermaschinen für Holzplatten. „Die Programmierumgebung LASAL lässt sich wirklich einfach erlernen. Die Applikation wird in funktionale, klar definierte Blöcke unterteilt. Der Anwender wird durch die grafische Darstellung und Hilfstexte unterstützt. Die grafische Darstellung veranschaulicht die Funktionsblöcke (realisiert mit OOP) und die Verbindungen zwischen den Blöcken. Diese Verbindungen funktionieren nach dem Client-Server-Prinzip, sie sind also ereignisorientiert. LASAL vereinfacht das Wiederverwenden des Codes sowie seine Erweiterung“, fasst Luca Vicentini zusammen.

Die Vorteile der OOP am Beispiel LASAL

- Abbildung von realen Maschinenkomponenten durch Softwareobjekte.
- Getestete Funktionsbausteine dank Kapselung – „use and forget“.
- Wiederverwendbarkeit der erstellten Funktionsbausteine und des Quellcodes.
- Klare, grafische sichtbare Schnittstellen nach außen. „Unsauberkeiten“ wie Zugriffe auf Daten an x Stellen im Projekt können somit gar nicht erst entstehen.
- Vereinfachte Zusammenarbeit in Teams.
- Durch die grafische Vererbung sind der Aufwand für Änderungen und Erweiterungen an bestehenden Funktionen minimal.
- Durch grafisches Bündeln mehrerer Funktionsbausteine kann eine komplexe Funktionsabfolge geschaffen werden. Beispiel: Zwei Objekte für ein Zahnrad und ein Objekt für einen Motor ergeben einen neuen Baustein „Getriebemotor“.
- Durch die Kapselung können die Bausteine einfach in Bibliotheken verwaltet werden.
- Abgeschlossene Bausteine schaffen die Möglichkeit, Projekte völlig ferngesteuert über Scripts (Python) zu erstellen oder abzuändern.
- Einfachere Lesbarkeit und Überprüfbarkeit führt zu gestiegener Qualität des Programmcodes.

SIGMATEK GmbH & Co KG
Sigmatekstraße 1, A-5112 Lamprechtshausen
Tel. +43 6274-4321-0
www.sigmatek-automation.com