Bild: Sigmatek GmbH & Co KG

Bild 1: Beim objektorientierten Ansatz wird der Quellcode gekapselt. Dies reduziert die Komplexität der Programme, erhöht die Übersichtlichkeit und macht die Maschinensoftware somit nachhaltig.

Modulares Software-Engineering

Bernhard Gangl, Abteilungsleiter Programmiertools bei Sigmatek, über objektorientiertes Engineering

Welche Trends in der SPS-Programmierung sind auszumachen? Unserer Ansicht nach gehen die Trends in Richtung eines Tools für alle Automatisierungsaufgaben, Modularisierung und Objektorientierung. All dies soll dann so benutzerfreundlich und einfach wie möglich umgesetzt werden.

Mit einem durchgängigen Softwaretool, das den gesamten Engineeringprozess – also neben der Steuerungsprogrammierung und Visualisierung auch Motion Control und Safety – abdeckt, wird eine hohe Entwicklungseffizienz erzielt. Zudem sollte ein modernes Tool alle Projektphasen unterstützen: Projektieren, Programmieren, Parametrieren, Diagnostizieren – dazu eine frühe Testbarkeit und Simulation.

Flexibilität und Modularisierung

Modulare Maschinenkonzepte ermöglichen Differenzierung und damit einen großen Wettbewerbsvorteil. Jeder Kunde hat sehr spezifische Vorstellungen, welche Funktionen seine Maschine zu erfüllen hat. Ausgehend von einer Basismaschine und Standard- bzw. optionalen Modulen stellt der Maschinenbauer die kundenindividuelle Variante zusammen und nimmt eventuelle Anpassungen vor. Softwareseitig steht der Maschinenhersteller vor der Herausforderung 'Wie minimiere ich den Programmieraufwand bei einer Vielzahl von Maschinenoptionen?'. Hier bieten objektorientierte Engineering-Konzepte eine optimale Lösung. Mit der objektorientierten Programmierung ist es möglich, neue Ausprägungen von Maschinenteilen mit minimalem Programmier- und Testaufwand umzusetzen. Sie ist somit der Schlüssel zur Modularisierung. Wie in der Mechanik, wo eine erprobte Konstruktion (mechatronische Einheit) wiederverwendet wird, können dank der modularen Struktur einmal erstellte und getestete Applikationsteile einfach wieder verwendet werden, ohne diese noch einmal überprüfen zu müssen. Software wird somit 'nachhaltig' und dies ist ein entscheidender Aspekt, da die Komplexität der Automatisierungsaufgaben immer noch zunimmt. Zudem lassen sich mit der objektorientierten Programmierung rund 30% der Engineeringkosten einsparen.

Objektorientierung

Grundprinzip ist eine durchgängige Modularität der einzelnen Programmblöcke von der untersten Ebene der einzelnen Funktionen bis hinauf zum Gesamtprojekt. Dadurch lassen sich selbst komplexe Programme strukturiert umsetzen und dann auch übersichtlich und flexibel halten. Eine 'Klasse' ist der Bauplan eines Objektes und definiert den Programmcode und die Datenelemente. Jede Klasse übernimmt bestimmte Aufgaben wie beispielsweise die Ansteuerung eines Antriebs oder eines Ventils. Beim objektorientierten Ansatz sind die Daten gekapselt und können von außen nicht verändert werden. Die gekapselten Objekte kommunizieren über Schnittstellen mit der 'Außenwelt'. Wenn diese Schnittstellen klar definiert sind, lassen sich Objekte einfach austauschen und es können mehrere Programmierer an einem Projekt arbeiten. Die getesteten Objekte lassen sich flexibel zusammenstellen und verbinden. Damit wird einerseits eine sehr effiziente Softwaregenerierung möglich und zudem steigt die Qualität der Software, was wiederum zu einer Performancesteigerung der Maschine bzw. Anlage führt.

Benutzerfreundlichkeit im Fokus

Als wir Ende der Neunziger mit der Entwicklung von Lasal starteten, hatten wir schon das Gesamtengineering im

Auge. Was muss ein einheitliches Tool bieten und wie lässt es sich einfach umsetzen? Wir haben mit Lasal ein System entwickelt, das grafisch unterstützt ist, wodurch die Software noch übersichtlicher wird. Die von Klassen erzeugten Objekte (Maschinenmodule) werden in Netzwerken dargestellt. Das heißt, der Code selbst ist nicht sofort ersichtlich. Dargestellt werden die Beziehungen der Programmteile zueinander sowie deren wichtigste Daten. So sieht der Entwickler mit einem Blick die Eigenschaften eines Maschinenteils und die Kommunikation mit anderen Maschinenteilen (Objekten). Der große Vorteil dabei ist, dass die Maschine in der Software grafisch nachgebildet wird. Das erleichtert die Umsetzung und hilft die Entwicklungs- und Wartungszeiten zu verkürzen. Zudem ist es Servicetechnikern so rasch und einfach möglich, eine Diagnose über die Fehlfunktion einer Maschine zu treffen. Wenn wir Lasal bei neuen Kunden präsentieren, gibt es immer einen Aha-Effekt, beim Blick auf ein Lasal-Netzwerk mit den Verschaltungen auf der grafischen Oberfläche zeigen. Da kommen Meldungen wie „Hallo, das ist genau das Blockschaltbild meiner Mechanik, die ich in der Maschine drinnen habe."

Tool unterstützt Anwender

Bei der Entwicklung von Lasal standen zwei Faktoren im Vordergrund: die Komplexität zu reduzieren und dem Benutzer ein komfortables Engineering Tool an die Hand zu geben, das ihn im gesamten Engineeringprozess unterstützt, also beispielsweise auch bei der Antriebsauslegung oder Simulation. Das 'all-in-one-Engineering' umfasst bei Sigmatek übrigens auch die Sicherheitstechnik, die sich mit den TÜV-zertifizierten Funktionen des Lasal Safety Designer ebenso integriert und intuitiv realisieren lässt. Wichtig ist: Der Anwender soll so schnell und komfortabel wie möglich zum Ziel kommen. Wir hatten von Anfang an den Anspruch, dem Anwender alle Vorteile der OOP zur Verfügung zu stellen, ohne dass er in der praktischen Anwendung mit der komplexeren Syntax der Objektorientierung in Berührung kommt. Lasal übernimmt inzwischen alles, was rundherum anfällt, wie z.B. die ganze Deklarierung der Klassen. Der Anwender kann sich auf die Implementierung der Methoden (Funktionen) konzentrieren. Der eigentliche Programmcode des Objektes wird in den gängigen Sprachen der IEC61131-3 (Strukturierter Text, Kontaktplan, SFC oder C) implementiert. Die Methoden der objektorientierten Programmierung stehen somit als durchgängige Erweiterung der vertrauten (Hoch)-Sprachen zur Verfügung. Die Visualisierungsbausteine sind direkt mit dem entsprechenden Funktionsbaustein verknüpft und können natürlich nicht nur in Lasal Class, sondern auch im Visualisierungsprojekt sofort verwendet werden.

Simulation spart Zeit und Kosten

Mit dem Lasal Runtime System (kurz LARS) kann der Code ohne angeschlossene Steuerung auf einem Windows-Rechner simuliert und getestet werden. Das gilt für komplette Applikationsprogramme, aber auch für noch nicht fertiggestellte Systemteile. So ist es möglich, die Funktionalität der Maschine zu überprüfen und den Quellcode zu debuggen, noch bevor die mechanischen Komponenten fertig sind. Eventuelle Designfehler lassen sich frühzeitig erkennen.

Integrierte Systemdiagnose

Eine frühe Testbarkeit der einzelnen Softwaremodule beschleunigt die Applikationsentwicklung. Der Entwickler wird mit effizienten Diagnose- und Debug-Tools wie Online-Debugger, Echtzeit Data Analyzer, Realtime-Trendaufzeichnungen, Online-Diagnose und umfassendem Projektvergleich unterstützt, die auch zu einer Verkürzung der Inbetriebnahme- und Servicezeiten beitragen. Diagnose- und Servicetools leisten einen wichtigen Beitrag zu einem fehlerlosen und unterbrechungsfreien Produktionsprozess. Denn die Kunden unserer Kunden erwarten eine einfache Integration und eine globale Wartung, die mit Fernwartungstools wie dem Lasal Remote Manager oder Bootdisk Manager komfortabel realisiert wird. Der Bootdiskmanger macht das Aktualisieren des Softwarestandes einer Maschine zum Kinderspiel: Einfach den USB-Bootstick konfigurieren, an die Steuerung anstecken und einen Reboot ausführen. Je nach Konfiguration lassen sich so Applikation, Visualisierung, Betriebssystem, Konfigurations- bzw. beliebige Dateien per Update-Script vom Bootstick auf die Steuerung übertragen, ohne dazu die tatsächlichen Programme auf den Maschinen kennen zu müssen.

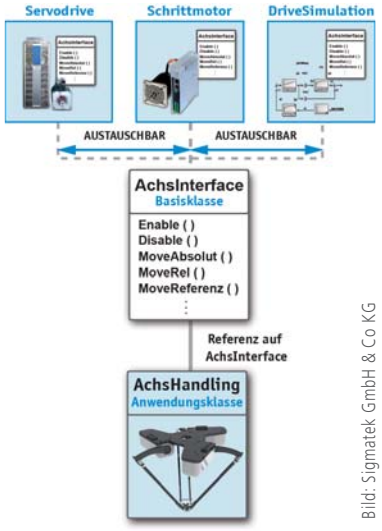

Bild: Sigmatek GmbH & Co KG

Bild 2: Mit der objektorientierten Programmierung wird Software modular – die Module lassen sich einfach wiederverwenden bzw. austauschen – das reduziert die Engineeringzeiten erheblich.

Programme automatisch erstellen

Das automatische Generieren und Anpassen der Maschinensoftware wird durch den Einsatz der Script-Sprache Python möglich. So können aus einem Basisprojekt verschiedene Maschinentypen modelliert werden. Dazu ein Beispiel: Ein Kunde bestellt den Basistyp A einer Maschine. Bei diesem Typ A hätte er gerne die Option 1, 3, 6 und 7. Bei Lasal ist es so, dass sich die Applikationssoftware für diese spezifische Maschine vollautomatisch erstellen lässt, ohne dass der Softwareingenieur irgendeine Programmzeile manuell verändern muss. Dazu wird auf eine Bibliothek mit vorgefertigten Klassen zugegriffen, die die verschiedenen Module enthalten. Basierend auf dem Grundprogramm werden die für Typ A spezifischen Module ausgewählt, die gewünschten Sonderfunktionalitäten der Optionen hinzugefügt, und dann wird quasi auf Knopfdruck das Systemprogramm erzeugt, das mit der Hardware interagiert. ■

www.sigmatek-automation.com