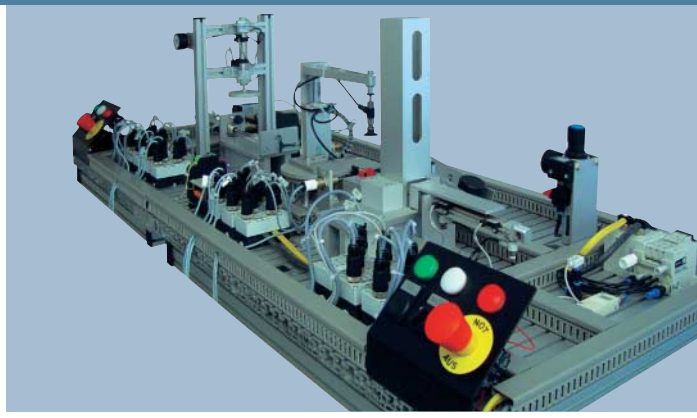
Studie zur Nutzung der Objektorientierung in der Automatisierungstechnik

Steigerung der Softwarequalität, Kostenersparnis durch Wiederverwendbarkeit von Softwarekomponenten und Modularisierung, die Verbesserung der Kommunikation zwischen verschiedenen am Entwicklungsprozess beteiligten Personengruppen und die durchgängige Verwendbarkeit von Werkzeugen und Methoden stellen ein noch nicht ausgeschöpftes Optimierungspotenzial bei der Erstellung von Automatisierungssoftware dar.

UML für die Steuerungsprogrammierung



Am Lehrstuhl für Prozessinformatik, an der Bergischen Universität in Wuppertal, wurde anhand einer prototypischen Implementierung beleuchtet, wie Objektorientierung und der Einsatz der „Unified Modeling Language" (UML) eingesetzt werden können um den Engineeringprozess zu verbessern. Als Automatisierungskomponenten kommen Beckhoff Hard- und Software zum Einsatz.

Der Forderung nach Wiederverwendbarkeit und Verbesserung der Softwarequalität wird in der Anwendungsentwicklung, außerhalb der Automatisierungstechnik, schon lange durch die Prinzipien und Methoden der Objektorientierung und ihrer Notation in der UML Rechnung getragen. Die Nutzung in der Automatisierungstechnik steht jedoch noch am Anfang.

Je nach Projektphase werden verschiedene Werkzeuge und Methoden verwendet, deren Daten beim Übergang von der einen in die andere Phase – im schlimmsten Falle – neu eingegeben werden müssen, weil keine geeigneten Schnittstellen der einzelnen Tools untereinander existieren. Die Idealvorstellung wäre ein übergeordnetes Werkzeug, welches sämtliche Informationen über das System in einem Modell konsistent bereitstellt und es ermöglicht, den Entwurf gleichermaßen auf einer abstrakten Ebene als auch in einer konventionellen Umgebung (z. B. E-CAE, IEC 61131-3) zu realisieren. Dies ist jedoch nur mit großem Aufwand zu verwirklichen. Mit vergleichsweise geringem Aufwand wurde ein Teil dieser Vorstellung in einem Prototypen umgesetzt.

Der Prototyp ermöglicht es, aus einem UML-Modell ein vollständiges und lauffähiges IEC 61131-3-Projekt samt Projektierungsinformationen für eine Beispielanlage zu generieren.

Prinzip der Codegenerierung an der Beispielanlage

Eine für die Modellierung von Embedded-Systemen konzipierte UML-Vorgehensweise wurde für die Automatisierungstechnik angepasst. Diese Anpassung berücksichtigt Hardware-Randbedingungen sowie Echtzeitaspekte und führt zu einem hierarchisch strukturierten Modell, das den Systementwicklungsprozess von einer umfassenden Anforderungsanalyse bis zum technischen Software-Entwurf pragmatisch unterstützt.

Als Modellierungswerkzeug wurde Real-Time Studio von Artisan verwendet. Dieses Tool eignet sich vor allem durch seine Flexibilität bei der Modellierung und seine offene Softwarearchitektur, die eine einfache Integration von Fremdtools mittels OLE-Anbindung zulässt. Auf Basis der o. g. Vorgehensweise wurde eine Beispielanlage mit diesem UML-Tool modelliert. Ein an der Bergischen Universität entwickelter Codegenerator erzeugt aus diesem Modell automatisch den IEC 61131-3-Code (Ablaufsprache und ST) und leitet Projektierungsinformationen ab. Der so aus dem UML-Modell gewonnene Code und die Projektierungsinformationen werden automatisch in die Programmierumgebung TwinCAT PLC bzw. in den TwinCAT System Manager importiert und automatisieren die Beispielanlage vollständig und korrekt.

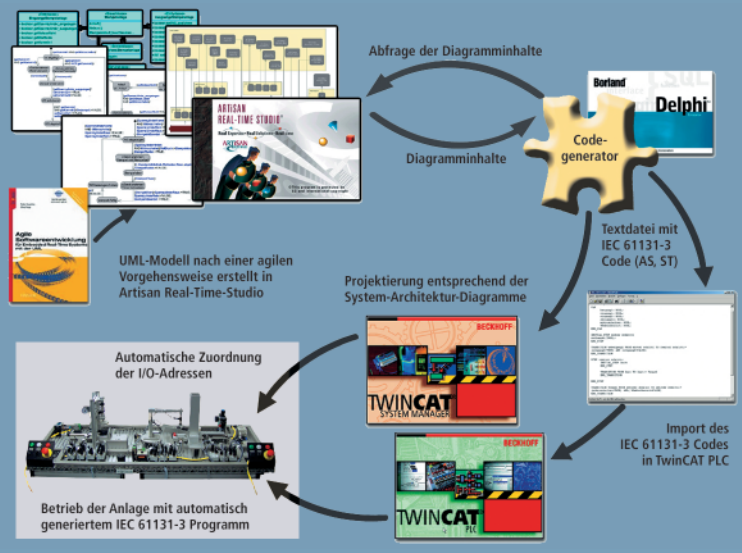
Die „Unified Modeling Language" (UML)

Die UML ist eine Sprache zur Spezifikation, grafischen Darstellung, Konstruktion und Dokumentation von objektorientierten Softwaresystemen, Geschäftsmodellen und Nicht-Softwaresystemen. Sie hat den Anspruch, eine allgemein verständliche Diskussionsbasis zwischen verschiedenen, am Projekt beteiligten, Personen beim Entwurf und der Entwicklung von Systemen zu bieten. Um Sachverhalte aus verschiedenen Perspektiven darstellen zu können, werden in der UML 1.x 9 Diagrammtypen definiert. Um die UML trotz ihrer Vielzahl an Diagrammen und ihrer Wahlfreiheit im Umgang mit diesen erfolgreich zu nutzen, ist der Einsatz einer Vorgehensweise und eines geeigneten UML-Tools notwendig. Die Vorgehensweise legt fest, welches Diagramm wann und wie anzuwenden ist.

Beckhoff kooperiert mit Hochschulen

Beckhoff unterstützt schon seit geraumer Zeit Arbeiten an verschiedenen Hochschulen und Fachhochschulen. Der Kern der automatischen Codegenerierung aus einem UML-Modell entstand als Bachelorarbeit von Daniel Witsch. Der Codegenerator wurde bereits weiterentwickelt. Dieses Thema wird auch Gegenstand zukünftiger Arbeiten sein.

Prinzip der Codegenerierung



Diese Beispielanlage stellt einen fertigungstechnischen Prozess dar, bei dem aufgrund einer Materialanalyse (induktiv, optisch) Werkstücke verschieden „weiterverarbeitet" werden. Helle und metallische Werkstücke werden auf der linken Anlagenseite gestempelt; Werkstücke aus dunklem Kunststoff werden aussortiert. Der Kran auf der rechten Anlagenseite transportiert die Werkstücke innerhalb der Anlage.

Das UML-Modell – Abbildung auf die IEC 61131-3

Bei der Modellierung wurden drei spezielle – für die Automatisierungstechnik relevante – Klassenarten verwendet: Entity-, Steuer- und Serviceklassen. Entityklassen dienen der zentralen Datenhaltung. Sie tragen viele Attribute und wenige oder keine Methoden. In einer Serviceklasse werden wiederverwendbare Funktionalitäten gesammelt. Dies können z. B. mehrfach benötigte Steueroperationen, Regler oder mathematische Funktionen sein. Steuerklassen übernehmen die zentrale Steuerung einer Task. Sie spielen innerhalb ihres Klassendiagramms eine Dirigentenrolle und bedienen sich der Variablen der Entityklassen und Methoden der Serviceklasse, um Aufgaben zu delegieren.

Im UML-Modell wird jede Task durch ein Paket (Package) dargestellt. Ein solches Paket enthält ein Klassendiagramm, welches aus Steuer-, Entity- und Serviceklassen zusammengesetzt ist. Dieses Klassendiagramm bildet das statische Modell der Task.

Jeder Methode einer Klasse wird ein Zustandsautomat zugeordnet (Abb. unten), der ihr dynamisches Verhalten definiert. Darüber hinaus wird der Steuerklasse an

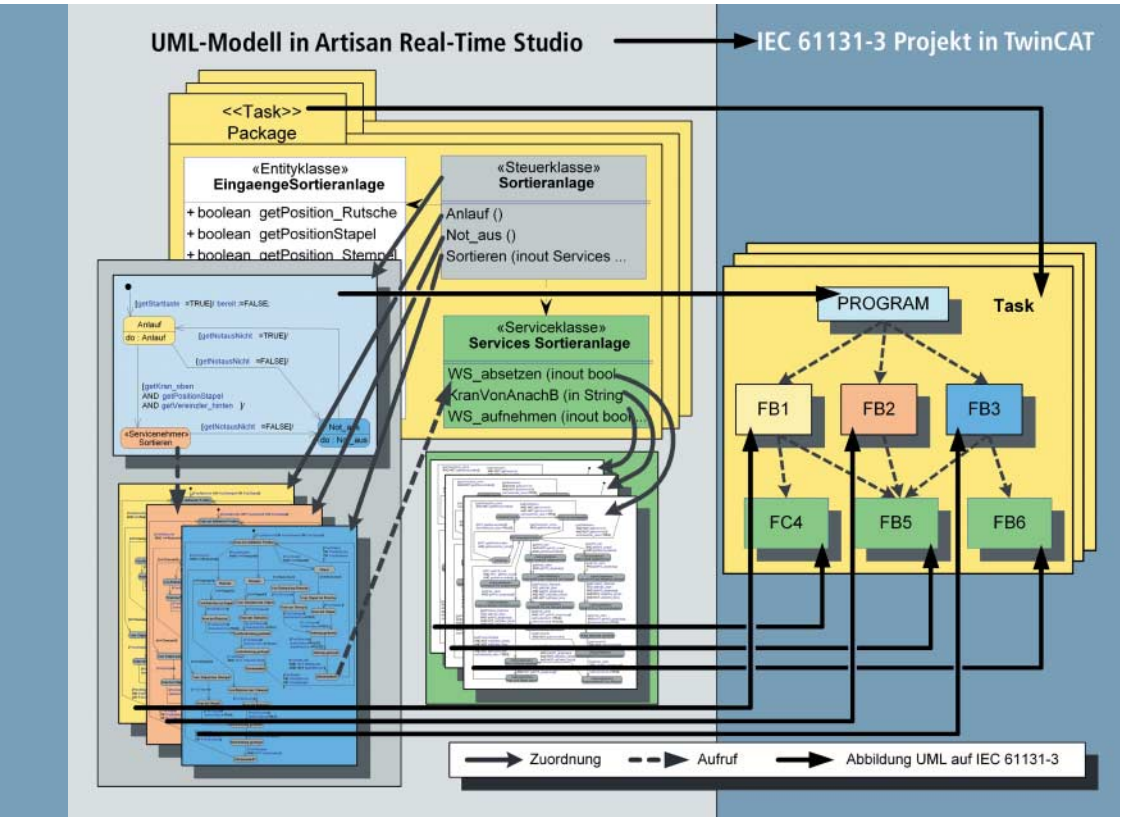
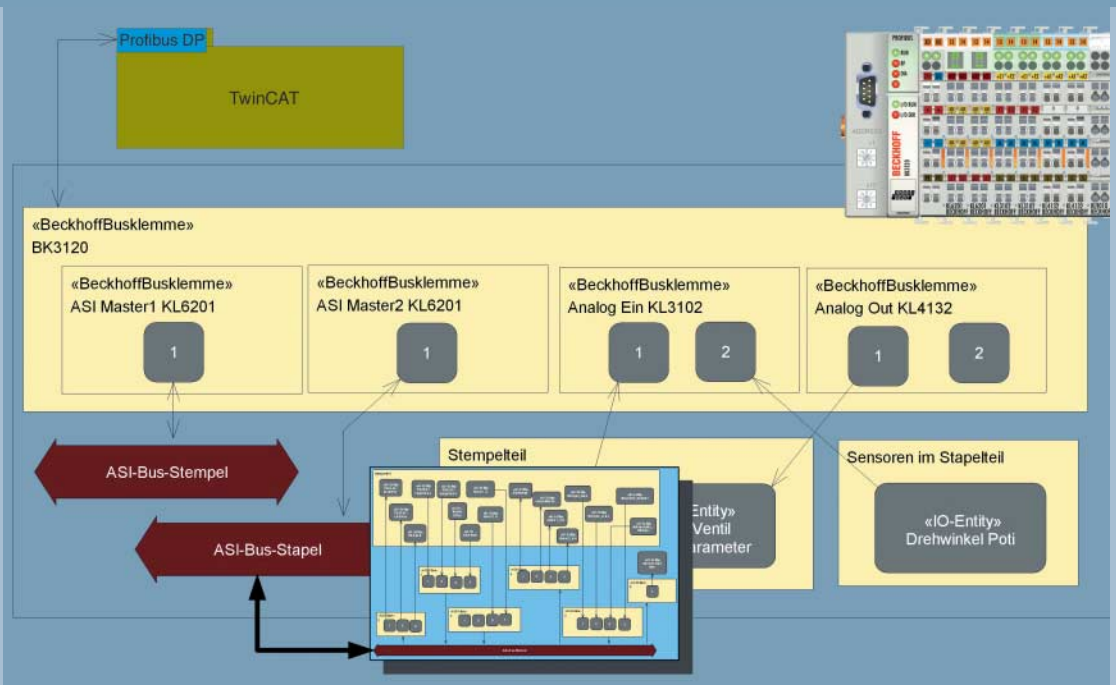


Abbildung des UML-Modells auf die IEC 61131-3



UML-Diagramm der Hardwarestruktur. An einen Bus angekoppelte Aktoren und Sensoren werden in einem separaten Diagramm dargestellt. Dadurch entsteht eine hierarchische Struktur.

sich ein Zustandsautomat zugewiesen, der das Zusammenspiel ihrer Methoden beschreibt. Dieser Zustandsautomat wird zum Hauptprogramm der Task. Das Hauptprogramm ruft die Methoden der Steuerklasse auf, die dann wiederum auf Methoden der Serviceklassen zurückgreifen können, um Standardoperationen durchzuführen. Auf diese Weise gliedert sich die Aufrufstruktur innerhalb der Task. Die Zustandsautomaten der UML werden in Ablaufsprache übersetzt.

Einbindung der Hardware

Artisan Real-Time Studio bietet für die Modellierung der Hardware sogenannte System-Architekturdiagramme (SAD). Diese entsprechen den Verteilungsdiagrammen in der UML. Mit Hilfe der SADs werden die vorhandenen Hardwarekomponenten und die Aktoren und Sensoren abgebildet. Die Verdrahtung der Aktoren und Sensoren mit den Ein- und Ausgängen der Steuerungshardware wird durch Verbindungspfeile dargestellt (Abb. oben). Außerdem werden die Variablen, die in den Klassen- und Zustandsdiagrammen verwendet werden, den Aktoren und Sensoren zugeordnet. Diese Informationen können bei einer Umverdrahtung automatisch an das Projektierungswerkzeug der TwinCAT-Umgebung übergeben werden, wodurch die Verbindung zwischen Hardware- und Softwareprojektierung gegeben ist.

Bei Umverdrahtung eines Sensors kann dies durch Umzeichnen eines Zuordnungspfeiles im SAD erfolgen. Die Umverdrahtung wird damit automatisch in der Software korrekt berücksichtigt.

Ausblick

Mit diesem Prototyp konnte gezeigt werden, dass mit Hilfe einer pragmatischen Anwendung der UML eine automatische Codegenerierung für die Automatisierungstechnik möglich ist. Aber noch kommen die eigentlichen Vorteile der Objektorientierung nicht zum Tragen. Die Vorteile – aber auch die Probleme – werden sich erst mit der Anwendung an einer komplexeren Anlage zeigen. Deshalb ist es nötig, dieses Verfahren bzgl. der Modellierung auf beliebig große Systeme zu erweitern und Mechanismen, wie Vererbung, zu berücksichtigen. Dies wird die Verwaltung von Varianten und Modulen deutlich vereinfachen. Hierzu liegen bereits Konzepte vor.

Ein Hauptargument gegen die Codegenerierung aus einem Modell ist der fehlende Rückweg. Es wird gefordert, dass z. B. der Inbetriebnehmer direkt im IEC 61131-3-Code Änderungen vornehmen kann und dass diese konsistent in das Gesamtmodell zurückfließen. Dies ist mit einer IEC 61131-3, so wie sie heute existiert, nur mit einer Vielzahl an Regeln möglich. Eine Erweiterung der IEC 61131-3 um Konstrukte der Objektorientierung wird bereits diskutiert. Sollten diese Konstrukte mit in die Norm aufgenommen werden, wäre eine Rückübersetzung und die Anwendung der UML ohne semantische Brüche möglich. Bis zu diesem Zeitpunkt ist der Ansatz einer gleichzeitigen Darstellung von UML-Modell und IEC 61131-3-Code eine konkrete Hilfestellung.

Autoren: Daniel Witsch und Univ.-Prof. Dr.-Ing. Birgit Vogel-Heuser

→ Lehrstuhl für Prozessinformatik, Bergische Universität Wuppertal
www.lfa.uni-wuppertal.de/uml2IEC61131