



Microsoft .net™

Michael Petig

.net und Echtzeit – kein Widerspruch!

Die Laufzeitumgebung CLR der .net-Technologie eignet sich auf Grund der fehlenden Echtzeit-Fähigkeit nicht für einen Einsatz in der Automatisierungstechnik – so die allgemeine Einschätzung. KW-Software hat nun einen Weg gefunden, um der Microsoft-Technologie den Weg in die Echtzeit-Welt zu ebnen.

Die Automatisierungstechnik bietet Laufzeitsysteme wie Sand am Meer. Zum einen gibt es für jeden Typ von Steuerung ein spezielles Laufzeitsystem: für die SPS-Steuerung, die Antriebssteuerung, die CNC-Steuerung, die Robotersteuerung, den Intelligenten Sensor/Aktor, die Schraubersteuerung und die Feldbus-Gateways. Diese Vielzahl multipliziert sich noch mit der Anzahl der Hersteller für einen Gerätetyp, da nahezu jeder Hersteller über ein eigenes Laufzeitsystem verfügt.

Mit der Vielzahl der Laufsysteme geht natürlich auch die Vielzahl der hierfür nötigen Software-Werkzeuge einher: Für

(Grafik: Computer & AUTOMATION)

jedes Gerät gibt es ein spezielles Programmiersystem oder Konfigurationswerkzeug.

Das Engineering in der Automatisierung orientiert sich heute also immer noch mehr an den Geräten und deren Technologien als an der eigentlichen Anwendung. Mit der Konsequenz, dass die Ausgaben für die Programmentwicklung immer stärker im Missverhältnis zu den Gesamtkosten eines Projektes stehen.

Die Eckpfeiler eines Laufzeitsystems

Die Kommunikations- und die Programmierschnittstelle bilden die zwei wesentlichen Eckpfeiler eines Laufzeitsystems. Was die Kommunikation der Geräte untereinander betrifft, laufen aktuell intensive Anstrengungen seitens der Hersteller, sich auf gewisse Standards zu einigen: Ethernet-Netzwerke stehen hierbei besonders im Rampenlicht.

Anders als bei der Kommunikation wird die Programmierschnittstelle von Geräten jedoch bisher kaum thematisiert. Dabei schränken die unterschiedlichen Codeformate, die auf die Automatisierungsgeräte geladen werden, die Flexibilität verteilter Systeme spürbar ein. Die Programme lassen sich selten entsprechend den funktionalen

Anforderungen des zu steuernden Prozesses optimal aufeinander abstimmen. Funktionalität und Leistungsfähigkeit der Geräte lassen sich so häufig nicht voll nutzen. – Der Schlüssel zu effektiv handhabbaren verteilten Systemen liegt folglich in einer vereinheitlichten Programmierschnittstelle.

Mit einer einheitlichen Programmierschnittstelle für alle Steuerungen, von der Antriebssteuerung bis zur SPS, wäre ein neuer Meilenstein in der Automatisierungswelt markiert. Dies umzusetzen, ist keine Utopie mehr: Im Zusammenwirken der .net-Technologie von Microsoft mit darauf aufbauenden Frameworks der Automatisierungstechnik lassen sich heute solche einheitlichen Schnittstellen generieren und damit die Vorteile verteilter Systeme ausnutzen.

Das .net-Laufzeitsystem CLR

Zu .net gehört ein Laufzeitsystem, die CLR (CLR steht für Common Language Runtime). Das heißt, die .net-Zielplattformen verfügen über ein einheitliches Laufzeitsystem für verschiedene Programmiersprachen. Dies ist eines der wesentlichen Merkmale von .net. Besonders vorteilhaft ist hierbei der einheitliche Zwischencode, für den die Begriffe MSIL (Microsoft intermediate language), CIL (common intermediate language, verwendet im ECMA-Standard) oder managed code stehen. MSIL ist die gemeinsame

Zwischencode-Basis für verschiedene Programmiersprachen. Die .net-Compiler für C++, MS Visual C# oder Visual Basic erzeugen also nicht mehr prozessorabhängigen Objektcode, sondern MSIL-Zwischencode. Liegt es da nicht nahe, auch MSIL für IEC-61131-Programme zu erzeugen?

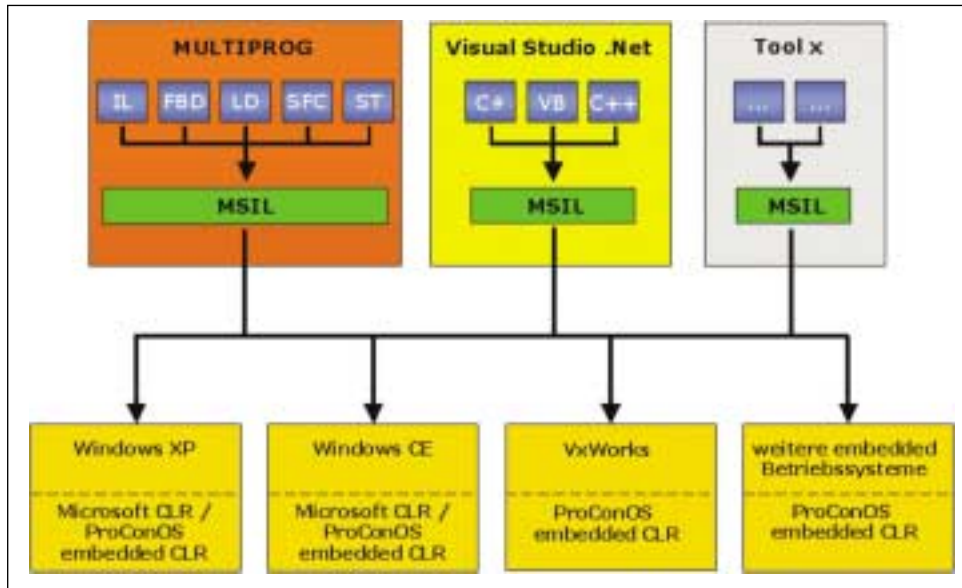
Die Sprache MSIL ist komplett offen, objektorientiert, Geräte-unabhängig und erlaubt jede Form von Mischungen, die dann auf das Gerät heruntergeladen werden. Ein steuerungstechnisches Anwendungsprogramm ließe sich zum Beispiel aus Teilen in C#, C++ und IEC 61131-3 zusammensetzen.

.net auch im Realtime-Bereich?

Bisher blieb die Anwendung der .net-Technologie auf das Engineering und auf nicht echtzeitkritische Prozesse beschränkt. Der Hintergrund dafür ist: Das .net-Laufzeitsystem CLR gilt als nicht echtzeitfähig.

KW-Software hat sich mit der Frage beschäftigt, ob sich das CLR-Laufzeitsystem nicht auch für Steuerungsaufgaben nutzen lässt.

Einer der zu prüfenden Gesichtspunkte war die Performance. Dabei zeigte sich, dass Microsoft CLR eine hohe Performance bei der Programmausführung aufweist. Dies wird erreicht durch einen Just-in-Time(JIT)-Compiler, der aus MSIL für die jeweilige CPU hoch optimierten,



Die Zukunft: MSIL und CLR stehen als Standardsprache zur Verfügung – Engineering-Werkzeug und Gerätesoftware sind nicht mehr zwingend miteinander gekoppelt. Beispielsweise ergibt sich so eine durchgängige Programmierung mit Multiprog, Visual Studio .net und anderen Tools, wie in der Grafik dargestellt.

direkt ausführbaren Code erzeugt. Die Ausführungsgeschwindigkeit ist vergleichbar mit der bisheriger Compiler, die direkt Maschinencode erzeugen. So verarbeitet eine Microsoft-CLR beispielsweise mit einem Pentium III bei 1 GHz 1000 BOOL-Verknüpfungen eines IEC-61131-Programms in 1,2 µs. Dieser Wert liegt damit genau in dem Bereich der Performance, die mit dem klassischen KW-Laufzeitsystem ProConOS erreicht wird. Ein weiterer wichtiger Aspekt war die

Echtzeit-Fähigkeit: Auf Grund der Garbage Collection wird der CLR eine Echtzeit-Fähigkeit abgesprochen. Die Garbage Collection ist Bestandteil der CLR und sorgt durch ein aktives Memory-Management dafür, dass der Speicher für nicht mehr benötigte Objekte automatisch aufgeräumt wird. Der Programmierer in den .net-Programmiersprachen wie C# braucht sich daher nicht mehr um die Speicherfreigabe zu kümmern. Nachteil ist: Solange die Garbage Collection läuft,

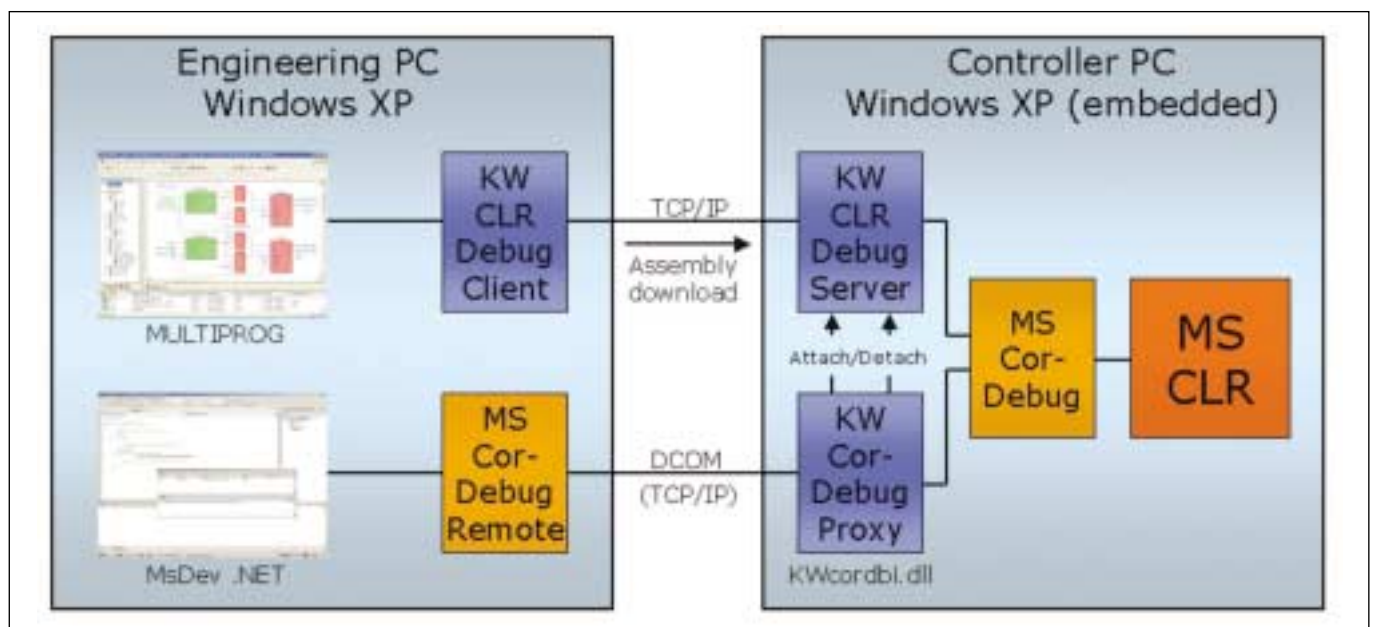
ist die Ausführung von Anwendungsprogrammen blockiert. Für Echtzeit-Programme bedeutet dies, dass der Jitter für die maximal zulässige Verzögerung der Programmausführung zu groß werden kann.

Die Rolle der Garbage Collection

Diese theoretische Einschränkung fällt in der Praxis jedoch weg, weil es bei Echtzeit-Programmen unüblich ist, Objekte zur Laufzeit zu kreieren, die anschließend nicht mehr benötigt werden. Bei der Abbildung von IEC-61131-Programmen in MSIL wird zum Beispiel der New-Operator nur beim Laden beziehungsweise Initialisieren des Programms benutzt, zur Laufzeit aber nicht. Für die Ausführung von IEC-61131-Programmen auf einer CLR wird also keine Garbage Collection benötigt. Sie beeinträchtigt daher auch nicht die Echtzeit-Fähigkeit.

Messungen des Jitters bei einer Microsoft-CLR haben ergeben, dass dieser sich im unteren ms-Bereich bewegt. Für typische SPS-Anwendungen mit Zykluszeiten oberhalb von 5 ms ist dies völlig ausreichend.

Die Voraussetzungen für den Einsatz von CLR als Laufzeitsystem für Steuerungen sind also erfüllt; die Nutzung einer einheitlichen Programmierschnittstelle auf Basis von MSIL für SPS-, CNC- und andere Technologie-Steuerungen ist möglich. Hierdurch erschließen sich völlig neue Möglichkeiten zur Mischung von



KW-Software hat Komponenten für die CLR-Umgebung entwickelt, mit denen sich beispielsweise Microsoft Visual Studio auch remote betreiben lässt. Alle Debug-Funktionen von Visual Studio sind so in dieser Umgebung verfügbar. (Bilder: KW-Software)

Programmiersprachen für Steuerungsanwendungen: Mit verschiedenen Tools wie Visual Studio und Multiprog erstellte Programmteile lassen sich beliebig mischen. Durch zusätzliche von KW-Software entwickelte Komponenten, lässt sich MS Visual Studio auch remote betreiben. Hierbei wird über eine TCP/IP-DCOM-Verbindung mit dem CLR-System kommuniziert. Es stehen dabei alle Debug-Funktionen von Visual Studio zur Verfügung. Paralleles, quasi zeitgleiches Debugging mit anderen Tools wie dem SPS-Programmiersystem Multiprog sind auch möglich.

Durch die Leistungsfähigkeit der CLR-Programmierschnittstelle lässt sich sogar die auf Automatisierungsgeräten üblicherweise vorhandene Firmware als MSIL-Programm laden. Die Notwendigkeit für ein proprietäres Laufzeitsystem entfällt. Ein SPS-Laufzeitsystem beispielsweise kann als integraler Bestandteil des Anwendungsprogramms in MSIL abgebildet werden. Lediglich spezifische Treiber-

schnittstellen müssen gegebenenfalls als unmanaged code implementiert werden. Zum Aufruf dieser Schnittstellen lassen sich Bibliotheksfunktionen in MSIL definieren, die dann SPS-typische Funktionalitäten zur Verfügung stellen.

CLR für Echtzeit-Betriebssysteme

Ein weiterer Aspekt ist, dass Microsofts CLR heute nur für Windows-Betriebssysteme zur Verfügung steht. Windows XP und CE haben sich zwar schon einen recht großen Marktanteil erobert, die vielen Echtzeit-Betriebssysteme anderer Anbieter sind jedoch aus der Embedded-Welt nicht wegzudenken. So beruht auch der Großteil der heutigen Steuerungen nicht auf Windows-Betriebssystemen, zumal für die verwendeten Prozessoren oftmals kein Windows-Betriebssystem existiert.

Die Frage lautet daher: Wie kann eine CLR für Embedded-Systeme eingesetzt werden?

Die Lösung heißt „ProConOS embedded CLR“ – das .net-Laufzeitsystem von KW-Software – entwickelt für die Embedded-Welt. Mit der ProConOS embedded CLR für das Echtzeit-Betriebssystem VxWorks können zum Beispiel mit Visual Studio erstellte C#-Programme auf dem VxWorks-System ausgeführt werden.

Das Echtzeit-Verhalten der Embedded-CLR wird dabei maßgeblich durch die Interruptlatenz- und Taskwechselzeiten des unterlagerten VxWorks bestimmt und liegt damit im μ s-Bereich.

Prinzipiell lässt sich die Embedded-CLR auf alle Betriebssysteme anpassen. So ist die ProConOS embedded CLR parallel neben einer MS-CLR auch auf Windows-Betriebssystemen einsetzbar.

Wobei klar festzuhalten ist, dass die Funktionalität der ProConOS embedded CLR insbesondere auf Steuerungsaufgaben in der Automatisierungstechnik zugeschnitten ist. So ist bei dem Produkt der Aspekt des geringen Speicherbedarfs



interview

„Jetzt wird die Gerätewelt offen“

KW-Software stellt zur SPS/IPC/Drives erstmalig eine Embedded-CLR für die Automatisierungstechnik vor. Andreas Orzelski, Geschäftsführer, erläutert die Bedeutung der Neuentwicklung.

□ Herr Orzelski, die CLR von .net eingesetzt als Laufzeitsystem: Wo sehen Sie die Grenzen für eine solche Vorgehensweise?

■ Orzelski: Wir glauben, dass .net die Softwarewelt auch in der Automatisierung verändern wird. Im Vergleich zu heutigen Lösungen wird .net eine viel engere Verbindung zwischen Automatisierungswelt und Bürowelt schaffen. Insofern sind Grenzen für mich nicht erkennbar. Technisch stößt die Microsoft-CLR im Bereich Online-Änderungen und Echtzeitfähigkeit an ihre Grenze. Hierfür bieten wir mit Embedded-CLR eine Lösung an.

□ Sie arbeiten an einer CLR-Version für Echtzeit-Betriebssysteme. Bis wann ist dies ein kauffähiges Produkt und wie sieht eine mögliche Vermarktung aus?

■ Orzelski: Dieses Jahr zeigen wir auf der SPS/IPC/Drives die neue Technologie Embedded-CLR und demonstrieren, dass Automatisierung mit .net machbar ist. Ein kauffähiges Produkt Embedded-CLR wollen wir nächstes Jahr auf der SPS/IPC/Drives vorstellen. Dieses neue Produkt Embedded-CLR wird Teil unserer offenen .net-Plattform sein. Die Embedded-CLR kann zusammen mit unserem IEC-61131-Programmiersystem MULTIPROG, mit der offenen Engineering-Plattform Automation Framework (Anm. d. Red.: siehe hierzu Special – „Funktionales Engineering“ der



Computer & AUTOMATION), aber auch stand-alone als Laufzeitsystem für eigene Anwendungen eingesetzt werden.

□ Welche Tragweite könnte diese Entwicklung bekommen?

■ Orzelski: Die .net-Technologie bietet die Grundlage für die verteilte Automatisierung, denn mit CLR und MSIL steht eine Standardsprache für Geräte zur Verfügung. Damit sind Engineering-Werkzeug und Gerätesoftware nicht mehr zwingend miteinander gekoppelt, sondern die Welt der Geräte wird offen. Automatisierungshersteller werden enorm flexibler, was die Durchgängigkeit und Funktionalität ihrer Geräte betrifft.

□ Können Sie sich diesbezüglich auch eine Zusammenarbeit mit anderen Anbietern von IEC-11631-Software vorstellen?

■ Orzelski: Wir setzen auf .net, weil es eine offene Plattform schafft. Durch die einheitliche Sprache MSIL ist unsere Embedded-CLR offen auch für andere Software. Es ist somit durchaus vorstellbar, dass Code von anderen IEC-61131-Werkzeugen generiert wird und auf der Embedded-CLR abgearbeitet wird.

berücksichtigt, um die .net-Technologie für Embedded-Laufzeitsysteme nutzbar zu machen.

Weiter lassen sich sogar Online-Änderungen durchführen. Online-Änderungen durchführen heißt, Programme und zugehörige Datenobjekte zu ändern, während diese sich in der Ausführung befinden. Die ProConOS embedded CLR ist in der Lage, die für eine Online-Änderung erforderlichen Maßnahmen auf der Basis eines geänderten MSIL-Programms autark zu erkennen. Hierdurch kann die Embedded-CLR Online-Änderungen unabhängig vom Programmierool

und der Programmiersprache unterstützen. So sind Online-Änderungen auch mit Visual Studio möglich. Bei einem C#-Programm kann zum Beispiel der Algorithmus geändert oder Variablen hinzugefügt werden, ohne dass die Ausführung des Programmes gestoppt werden muss. hap



Michael Petig

ist bei KW-Software als Development Manager für die Entwicklung von SPS-Systemsoftware, insbesondere für Laufzeitsysteme verantwortlich.