OOP-basierte Implementierung eines erweiterbaren Skriptinterpreters in LabVIEW

Peter Schwarz
A.M.S. Software GmbH, Ellerau

Kurzfassung

In einigen Projekten wünschen unsere Kunden neben der Verwendung von LabVIEW zusätzlich eine per textorientierter Programmierung änderbare Steuerung von Mess- und Prüfabläufen. Zu diesem Zweck verwendet AMS schon seit Jahren die Skriptsprache AMSL. AMSL ist dabei über dynamisch geladene VIs um in LabVIEW implementierte Funktionen erweiterbar.
Im Zusammenhang mit cRIO-Systemen gibt es ähnliche Anforderungen für LV-Realtime-Anwendungen. Um diese erfüllen zu können, haben wir AMSL nativ in LabVIEW implementiert und dabei objektorientierte Programmierung verwendet.
Das AMSL-Skript wird vor seiner Ausführung komplett analysiert. Im Artikel wird der grundsätzliche Aufbau dargestellt und ein Ausblick auf mögliche weitere Entwicklungen gegeben.

Abstract

In addition to the use of LabVIEW, in some projects our customers also want a control system for measuring and test sequences that can be modified by text-orientated programming. AMS has used the script language AMSL for this purpose for several years now. At the same time, AMSL can be expanded with functions implemented in LabVIEW using dynamically loaded VIs.
In connection with cRIO systems, LV Real-Time applications have similar requirements. To be able to meet these, we have implemented AMSL natively in LabVIEW and used object-orientated programming when doing so.
The AMSL script is completely analyzed before its execution. The article shows the basic structure and offers an insight into possible further developments.

Die Idee

In einigen Projekten ist es nötig und erwünscht, Prüfabläufe aus einzelnen Schritten bzw. Teilschritten zusammenzusetzen. Dieses Zusammensetzen könnte im Prinzip auch durch ein entsprechendes LabVIEW-Programm erfolgen, dies kann aber je nach Realisierung die Notwendigkeit einer Neucompilierung der LabVIEW-Applikation bedeuten. Den eigentlichen Ablauf mit einer Skiptsprache zu realisieren kann hier helfen, Anpassungen schneller und gut archivierbar vorzunehmen.
Der hier gewählte Ansatz erlaubt es zudem, die Programmiersprache mit Hilfe von LabVIEW einfach um zusätzliche Befehle/Funktionen zu erweitern, so dass komplexe und zeitkritische Prüfschritte als ein Skriptbefehl ausgeführt werden können. Dadurch, dass diese Erweiterungen auch dynamisch hinzugeladen werden können, ist es möglich, den gleichen Programmrahmen zur Steuerung beliebiger Prüfungen und Prüfplätze zu nutzen. Dadurch, dass AMLS in LabVIEW unter Einsatz objektorientierter Programmierung implementiert wurde, wird neben Windows als Zielsystem z.B. auch die Programmierung von LabVIEW Real-Time-Systemen möglich.

AMSL (AMS Scripting Language)

AMSL ist eine typsichere Skriptsprache, die gleichermassen an C und PASCAL erinnert. In AMSL sind Groß- und Kleinschreibung nicht signifikant.

Ein einfaches Skript soll hier als Beispiel dienen:

```
function Main(args:string[]):int
  DLG.Messagebox("Titel", "Hello world")
  return 0
endfunc
```

Die Funktion „Main“ ist immer der Startpunkt der Ausführung eines Skriptes und kann (wenn erwünscht) eine beliebige Anzahl von Parametern als String-Array erhalten.
Der dann folgende Aufruf der externen Funktion „DLG.Messagebox“ stellt die Verwendung einer Erweiterungsfunktion vor, die die Aufgabe hat, eine Messagebox darzustellen.
Der Return-Befehl legt schliesslich den Rückgabewert der Funktion fest.
Der Anwender kann beliebige Funktionen selbst implementieren und diese entsprechend auch aufrufen. Rekursion ist dabei ebenfalls möglich.
AMSL hat insgesamt folgende Sprachelemente:

- Funktionsdeklaration
- Variablendeklaration

- For-, While- und Repeat-Schleifen
- If-Then-Else- und Case-Strukturen
- Zuweisungen, Berechnungen
- String- und Datei-Funktionen
- Include- und Import-Anweisungen
- Funktionsaufruf mit Call-By-Value und Call-By-Reference
- Rekursion ist möglich

Implementierung

Bei der Implementierung wurde zunächst die lexikalische Analyse implementiert, die den Programmquelltext in Tokens zerlegt, welche die erkannte Art des Textes und die Quellcodepositionsinformationen enthalten. Aus dieser Liste wird dann vor Ausführung eine Liste von AMSL-Skriptobjekten erzeugt, die dann beim Ausführen des Skriptes mittels einer Excute-Methode ausgeführt werden.

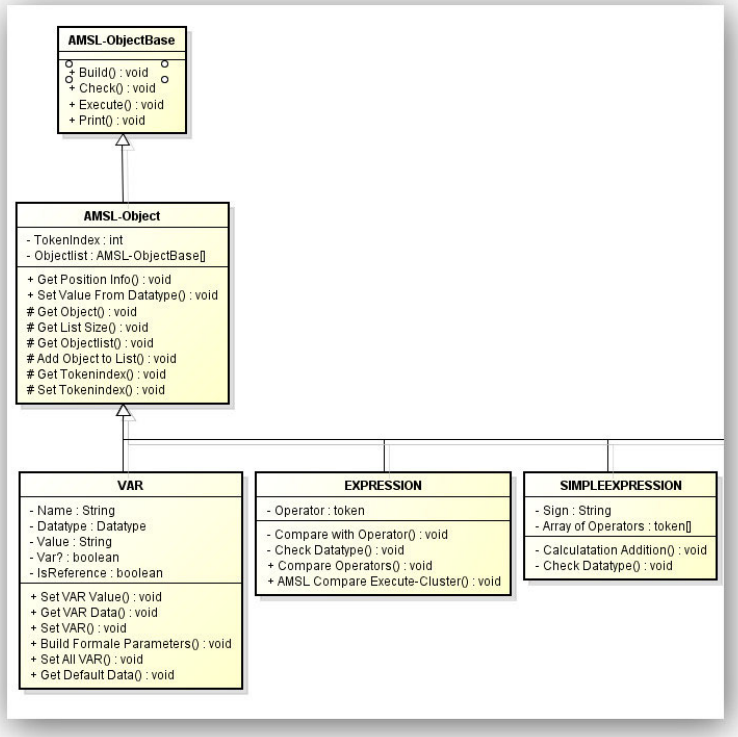

Bild 1: Ausschnitt aus der AMSL-Klassenhierarchie

Durch entsprechend gekapselte Laufzeitumgebungen ist es möglich, auch mehrere Skripte parallel laufen zu lassen. In Zukunft könnte eine Kommunikation zwischen Skripten implementiert werden. Ebenso ist es denkbar, dass ein Skript eine oder mehrere seiner Funktionen als nebenläufigen Prozess startet.
Bereits heute steht eine moderne Entwicklungsumgebung mit einem die Sprachsyntax unterstützenden Editor zur Verfügung, aus der auch Debugging möglich ist.

Zusammenfassung

Mit AMSL steht uns und unseren Kunden eine flexible texorientierte Möglichkeit der Zusammenstellung von Prüfabläufen zur Verfügung. Die Implementierung mittels objektorientiertem LabVIEW hat sich dabei bewährt.

[1] Virtuelle Instrumente in der Praxis – Begleitband zur VIP2004/Seite 168: Datenbankgestütztes Prüfsystem mit LabVIEW für ein Miele-Prüffeld

[2] Virtuelle Instrumente in der Praxis – Begleitband zur VIP 2011/Seite 80: Interpreter für skriptgesteuerte Prüfabläufe auf cRIO/LabVIEW im Labor bei Miele im Werk Lehrte