LabVIEW

LabVIEW ist ein grafisches Programmiersystem von National Instruments. Das Akronym steht für „**Lab**oratory **V**irtual **I**nstrumentation **E**ngineering **W**orkbench“.

Die erste Version erschien 1986 für Macintosh-Computer. Heute gibt es die Entwicklungsumgebung außerdem für Windows und Linux. Vergleichbar entwickelte Hewlett-Packard (inzwischen Agilent) die visuelle Programmiersprache VEE. Hauptanwendungsgebiete von LabVIEW sind die Mess-, Regel- und Automatisierungstechnik. Die Programmierung erfolgt mit einer grafischen Programmiersprache, genannt **„G“**, nach dem Datenfluss-Modell. Im Vordergrund steht dabei die Datenerfassung und -verarbeitung. LabVIEW-Programme werden als Virtuelle Instrumente oder einfach VIs bezeichnet. Sie bestehen aus zwei Komponenten: Das Frontpanel enthält die Benutzerschnittstelle, das Blockdiagramm den grafischen Programmcode. Dieser wird nicht von einem Interpreter abgearbeitet, sondern compiliert. Dadurch ist die Leistung vergleichbar mit anderen Hochsprachen. LabVIEW benutzt die gleichen Bibliotheken und Datenerfassungsmodule wie LabWindows/CVI, der Integrierten Entwicklungsumgebung der Firma National Instruments und ist deshalb kompatibel mit LabWindows/CVI. Für viele komplexe mathematische Aufgaben stehen auch Funktionsbibliotheken zur Verfügung. Ähnlich wie Matlab deckt LabVIEW auch die Bereiche SPS-Steuerung und flexible Versuchsautomatisierung ab.

LabVIEW

Entwickler	National Instruments
Aktuelle Version	LabVIEW 2017 (Mai 2017)
Betriebssystem	macOS, Windows, Linux
Kategorie	Programmiersprache
Lizenz	proprietär
deutschsprachig	ja

LabVIEW (http://www.ni.com/labview/d/)

Inhaltsverzeichnis

Programmiermethode

Funktionsblöcke werden in LabVIEW (genau wie vollständige Programme) als Virtuelle Instrumente (VIs) bezeichnet. Dies kommt daher, dass prinzipiell jedes Programm als Unterprogramm (Sub-VI) in einem anderen verwendet werden kann, bzw. jedes Sub-VI auch eigenständig lauffähig ist. Aufgrund des Datenfluss-Konzeptes waren bis zu Version 8.6 rekursive Aufrufe grundsätzlich nicht möglich. Mit zusätzlichem Aufwand ließen sich aber auch Rekursionen verwirklichen.[1] Ab Version 9.0 kann ein ablaufinvariantes VI sich selbst als SubVI enthalten und damit rekursiv aufrufen.[2]

Der Programmierer verbindet VIs mit Verbindungslinien (Drähten) und definiert damit den Datenfluss. Jedes VI kann dabei Ein- und Ausgänge besitzen. Die Ausführung eines VIs beginnt, wenn alle Eingangsdaten vorhanden sind; die Ergebnisse liegen erst dann an den Ausgängen an, wenn das gesamte Unterprogramm abgearbeitet ist. Auf diese Weise wird die Abarbeitungsreihenfolge der Schritte durch Datenabhängigkeiten definiert. Eine vordefinierte Reihenfolge (z. B. „von rechts nach links“) gibt es nicht.

Besitzt ein Sub-VI keine Eingänge, wird es bei Programmstart ausgeführt. Besitzt es keine Ausgänge, werden die Ergebnisdaten entweder verworfen oder auf einem anderen Weg „verwertet“ (z. B. Schreiben auf Festplatte oder Netzwerk, Ausgabe auf Peripheriegeräte). Genauso kann ein Sub-VI ohne Eingänge Daten von Peripheriegeräten erhalten oder selbst generieren (z. B. per Zufallsgenerator).

Sub-VIs können beliebig tief verschachtelt werden. Viele der LabVIEW-eigenen Funktionen sind ihrerseits normale VIs, die auch vom Programmierer bearbeitet werden können (wenngleich dies in der Regel nicht zu empfehlen ist). Letztlich basieren alle VIs auf einer Reihe grundlegender Funktionen, sogenannter Primitive, die sich nicht als VIs öffnen lassen.

Viele VIs und Primitive in LabVIEW sind polymorph, d. h. ihre Funktionalität passt sich an den Typ der übergebenen Daten an. Beispielsweise kann die Build-Array Funktion für die Erstellung jeglicher Felder genutzt werden, d. h. Strings, Integer oder auch Arrays und Cluster. Es ist auch möglich, eigene polymorphe VIs zu erstellen. Letztlich handelt es sich hierbei um eine Sammlung mehrerer VIs mit unterschiedlichen Datentypen an den Ein- und Ausgängen.

Datenquellen und Datensenken können mit Anzeige- und Bedienelementen auf dem Frontpanel verknüpft sein. So kann z. B. eine Zahleneingabe mit einem Drehknopf und eine Ausgabe einer booleschen Variablen mit einer Leuchtdiode realisiert werden.

Bei sehr großen und umfangreichen Projekten ist es wie in anderen Programmiersprachen wichtig, von Anfang an eine durchdachte Struktur zu verwenden und den Code zu modularisieren. Durch den vorhandenen Projektmanager (ab V8.0) wird dies unterstützt. Die Verwaltung einer großen Anzahl an VIs sowie externer Dateien wird dadurch übersichtlicher. Auch die Versionsverwaltung gestaltet sich hiermit einfacher. Eine wesentliche Neuerung (ab V8.20) besteht darin, objektorientiert programmieren zu können. Klassen und Attribute sowie deren Methoden können dabei auch vererbt werden.

LabVIEW Robotics 2009 enthält Werkzeuge für den Entwurf eines Robotersystems. Teil des Softwarepakets ist das Robotics Module, das eine umfassende Robotikbibliothek mit Anbindungsmöglichkeiten an Standard-Robotiksensoren und -aktoren, grundlegenden Algorithmen für den intelligenten Betrieb sowie Wahrnehmungs- und Motorsteuerungsfunktionen für Roboter und autonome Fahrzeuge umfasst.

LabView, TCP/IP und XML Webdienste

TCP/IP Sockets übertragen Zeichenketten in LAN und Internet. Sie werden zum Beispiel für das HTTP Protokoll verwendet welches Websites im WebBrowser darstellt. LabView macht im LAN Gebrauch von TCP/IP Sockets und das für unterschiedliche Zwecke:

- Funktionen unter *Datenkommunikation->Protokolle->TCP*.
- Webbrowser-OLE auf dem Frontpanel einbinden.
- Auslesen von Messwerten, z. B. von DMM und NWA - Geräten (siehe Measurement und Automation Explorer).
- Fernsteuerung über Remote Panel.
- XML Web Dienst.

LabView spezifisch ist dabei einerseits die Fernsteuerung über das Remote Panel. Dabei kann ein Benutzer sich über den Webbrowser (IE bevorzugt unterstützt) mit einem geöffneten Frontpanel verbinden. Das Frontpanel selbst gibt die Steuerung dann ab. Der beabsichtigte Nutzeffekt ist der Zugang über einen zweiten PC. Jedoch muss auf dem Quell PC ein Fenster geöffnet sein. Weiter verlangt NI für Nutzeranzahlen größer als eins zusätzliche Lizenzgebühren.

LabView spezifisch sind andererseits selbst erstellbare XML Webdienste. Sie sind erst in der Full-Version zum Preis von etwa 3300,-€ enthalten. Diese Webdienste benötigen kein GUI-Fenster auf dem Host-PC. Ein Beispiel-Howto zeigt wie der Nutzer einen HTTP-GET-Request mit zwei Parametern an den Service sendet. Die Parameter sind dabei in der URL enthalten. Der Dienst führt ein VI aus und antwortet mit einer XML-Datei. Die URL-Parameter werden in dem zu implementierenden Dienst als Controls in dem VI erstellt - alle Indikatoren des VIs erscheinen mit ihren Werten in der rückgesendeten XML Datei. Ein Webdienst könnte man also anwenden um Aktuatoren zu steuern und Messwerte zu erfassen. Mit dem Webdienst kann man eigene grafische Oberflächen implementieren und eigentlich jedes Client Betriebssystem nutzen. Auch Batch-Skripte für lange Versuchsabläufe wären möglich.

Um ein TCP Service Socket implizit bereitzustellen gibt es einerseits die Möglichkeit es über den Projektbaum einer ausführbaren EXE beizulegen. Fall man eine Anwendung mit Remote Panel startet, so ist zur Laufzeit das Socket vorhanden.

Die andere Möglichkeit der Bereitstellung ist der NI Webserver und sein Dokumentroot Verzeichnis z. B. unter *C:\Program Files (x86)\National Instruments\Shared\NI WebServer\www*. Der Dienst erscheint in der Dienste Konsole unter NI APPLICATION WEBSERVER und lauscht auf Port 8080. Nutzt man den Server auf einem Host mit der LabView Entwicklungsumgebung so kann man den Service über den Projektbaum publizieren. Nutzt man die LabView Laufzeitumgebung auf einem anderen Host so kann man den Service über einen Installer bereitstellen den man auf dem Entwicklungssystem über den Projektbaum erstellt.

Vorteile

- Eine wichtige Konsequenz LabVIEWs grafischer Programmierung ist die Einfachheit, mit der in LabVIEW parallele Abläufe programmiert werden können. Es reicht, zwei Sub-VIs ohne Datenabhängigkeit nebeneinander zu legen, um sie gleichzeitig mit Multithreading abzuarbeiten. Man muss allerdings, ähnlich wie in text-basierten Programmiersystemen, auf mögliche Race Conditions achten und, wo nötig, Ressourcen sperren. Zur Synchronisierung bzw. Kommunikation zwischen mehreren Threads stehen verschiedene Möglichkeiten zur Verfügung (z. B. Semaphoren, Melder, Warteschlangen).

- Das Frontpanel von LabVIEW ist ein sehr bequemes Mittel, um Programme mit guter grafischer Bedieneroberfläche zu erstellen. Bei allen Programmierarbeiten in LabVIEW muss der Programmierer prinzipiell keinen Text eingeben, außer Beschriftungen von Gestaltungselementen.

- Die graphische Darstellung des Programmablaufs erhöht zumindest bei nicht zu umfangreichen Vorhaben die Lesbarkeit deutlich. Insbesondere Naturwissenschaftler und Techniker verstehen die Programmlogik meist recht schnell und können Software damit an ihre konkreten Bedürfnisse anpassen.
- Die je nach Lizenz mitgelieferten umfangreichen Funktionsbibliotheken decken insbesondere die Datenanalyse und Mathematik sehr weitgehend ab. Aber auch die Ansteuerung von zusätzlichen (auch externen) (Mess-)Geräten und Systemfunktionen ist gut gelöst.
- Über die unterstützten Kommunikationsprotokolle und Verbindungstechniken ist es möglich, auch weit entfernte Geräte (z. B. an unzugänglichen Stellen oder in anderen Ländern) zu steuern und zu nutzen. Hier kommt unter anderem TCP zum Einsatz.
- LabVIEW ermöglicht ab der Version 8.6.1 auch die Programmierung von Mikrocontrollern und DSPs, die in von National Instruments produzierten Messgeräten enthalten sind. Es unterstützt auch einige Echtzeitbetriebssysteme.
- Ab Version 2009 bietet LabVIEW die Möglichkeit der parallelen Programmierung von Multicore-Prozessoren und FPGAs und den Zugriff auf Wireless-Technologien.

Nachteile

Neben den genannten Vorteilen hat die graphische Programmierung gegenüber der textbasierten auch Nachteile:

- LabVIEW-Programme lassen sich nur mit der originalen LabVIEW-Entwicklungsumgebung bearbeiten, an deren Funktionsumfang man gebunden ist. Allerdings lassen sich Funktionen aus dynamischen Bibliotheken oder ActiveX-Objekte nutzen. Der Plattformunabhängigkeit geschuldet sind Hindernisse in der Gestaltung von Benutzeroberflächen, so werden Windows-Hotkeys nicht unterstützt, und das Verhalten von Accelerator-Keys entspricht nicht exakt dem Verhalten des Betriebssystems (hier: Fokus-Verlust). Die Unicode-Unterstützung ist unzureichend.
- Ausführbare LabVIEW-Programme können vom Entwicklungssystem zwar erstellt werden, erfordern jedoch die Installation einer Laufzeitumgebung auf dem Zielsystem (vergleichbar mit der Installation des .NET-Frameworks für .NET Applikationen). Bei Verwendung von bestimmten Zusatzmodulen, wie z. B. IMAQ Vision, ist zudem eine kostenpflichtige Lizenz pro Zielplattform notwendig.
- Die Prinzipien moderner Objektorientierung versucht National Instruments mit neueren LabVIEW Versionen zwar nachzubilden, jedoch gelingt dies bisher nur unzureichend. Zudem zeigt die Programmierung gegen große bestehende Klassenhierarchien wie das Microsoft .NET-Framework die Grenzen im Umgang mit grafischen Zugriffsknoten auf – mit textbasierten Programmiersprachen, zum Beispiel C#, sind dieselben Aufgaben i. d. R. schneller programmierbar.
- Kleine Änderungen können aufwendige Neustrukturierungen nach sich ziehen, wenn das Schaffen von Raum auf dem Blockdiagramm durch Verschieben geschieht, da dann die Drähte und Symbole oftmals neu geordnet werden müssen, um die Übersichtlichkeit wiederherzustellen. Dieses Problem kann jedoch durch Strukturierte Programmierung gemildert werden (insbesondere durch konsequente Verwendung von Sub-VIs).

- Der einfache Einstieg in die LabVIEW-Programmierung verleitet dazu, die ordentliche Planung des Projektes zu vernachlässigen.

Versionen

Name/Version	Build Nummer	Datum
Start des LabVIEW Projekts		April 1983
LabVIEW 1.0 (für Apple Macintosh)		Oktober 1986
LabVIEW 2.0		Januar 1990
LabVIEW 2.5 (Erste Version für Sun & Microsoft Windows)		August 1992
LabVIEW 3.0 (Multiplattform)		Juli 1993
LabVIEW 3.0.1 (Erste Version für Windows NT)		1994
LabVIEW 3.1		1994
LabVIEW 3.1.1 (Erste Version mit „application builder“)		1995
LabVIEW 4.0		April 1996
LabVIEW 4.1		1997
LabVIEW 5.0		Februar 1998
LabVIEW RT (Real Time)		Mai 1999
LabVIEW 6.0 (6i)	6.0.0.4005	26.7.2000
LabVIEW 6.1	6.1.0.4004	4.12.2001
LabVIEW 7.0 (Express)	7.0.0.4000	April 2003
LabVIEW PDA module (Erste Version)		Mai 2003
LabVIEW FPGA module (Erste Version)		Juni 2003
LabVIEW 7.1		2004
LabVIEW Embedded module (Erste Version)		Mai 2005
LabVIEW 8.0		September 2005
LabVIEW 8.20 (Objektorientierte Programmierung)		August 2006
LabVIEW 8.2.1	8.2.1.4002	21.2.2007
LabVIEW 8.5	8.5.0.4002	2007
LabVIEW 8.6	8.6.0.4001	24.7.2008

LabVIEW 8.6.1	8.6.0.4001	10.12.2008
LabVIEW 2009 (32 und 64-bit)	9.0.0.4022	4.8.2009
LabVIEW 2009 SP2	9.0.1.4011	8.1.2010
LabVIEW 2010	10.0.0.4032	4.8.2010
LabVIEW 2010 f2	10.0.0.4033	16.9.2010
LabVIEW 2010 SP1	10.0.1.4004	17.5.2011
LabVIEW für LEGO MINDSTORMS (2010 SP1)		August 2011
LabVIEW 2011	11.0.0.4029	22.6.2011
LabVIEW 2011 SP1	11.0.1.4015	1.3.2012
LabVIEW 2012	12.0.0.4029	August 2012
LabVIEW 2012 SP1	12.0.1.4013	März 2013
LabVIEW 2013	13.0.0.4047	August 2013
LabVIEW 2013 SP1	13.0.1.4017	März 2014
LabVIEW 2014	14.0.0.4036	August 2014
LabVIEW 2014 SP1		März 2015
LabVIEW 2015		August 2015
LabVIEW 2015 SP1		März 2016
LabVIEW 2016		August 2016
LabVIEW 2017		Mai 2017

Literatur

- Wolfgang Georgi; Ergun Metin: *Einführung in LabVIEW*, Fachbuchverlag Leipzig im Carl Hanser Verlag, 4. neu bearbeitete Auflage 2009, ISBN 978-3-446-41560-7
- Bernward Mütterlein: *Handbuch für die Programmierung mit LabVIEW. Mit Studentenversion LabVIEW 8 (Gebundene Ausgabe)*, 1. Auflage April 2007, ISBN 3-8274-1761-9
- Kurt Reim: *LabVIEW-Kurs : Grundlagen, Aufgaben, Lösungen*, Vogel Buchverlag, 1. Auflage 2014, ISBN 978-3-8343-3294-3
- Sebastian Trella; Thorsten Leimbach: *Roberta - Programmieren mit LabVIEW*, Fraunhofer Verlag, 1. Auflage Mai 2014, 118 S., ISBN 978-3-8396-0692-6

Weblinks

- Handbuch (http://info.hit-karlsruhe.de/info-ss09/e_kart_batterieman/data/Lab%20View%20Handbuch.pdf)
- LabView und Webtechnologien (http://www.g-objectview.de/dokumente_download/produkte/javapanel/javapanel_diplom.pdf)

Einzelnachweise

1. Rekursion in LabVIEW (english) (http://digital.ni.com/public.nsf/allkb/7140920082C3AC15862572840015A81E)
2. Verwendung der Rekursion bei VIs (english) (http://zone.ni.com/devzone/cda/tut/p/id/9387)

Abgerufen von „https://de.wikipedia.org/w/index.php?title=LabVIEW&oldid=168059498“

- Diese Seite wurde zuletzt am 11. August 2017 um 10:42 Uhr bearbeitet.
- Der Text ist unter der Lizenz „Creative Commons Attribution/Share Alike“ verfügbar; Informationen zu den Urhebern und zum Lizenzstatus eingebundener Mediendateien (etwa Bilder oder Videos) können im Regelfall durch Anklicken dieser abgerufen werden. Möglicherweise unterliegen die Inhalte jeweils zusätzlichen Bedingungen. Durch die Nutzung dieser Website erklären Sie sich mit den Nutzungsbedingungen und der Datenschutzrichtlinie einverstanden. Wikipedia® ist eine eingetragene Marke der Wikimedia Foundation Inc.