

R. Jamal: „Konfigurieren UND Programmieren – ein Paradoxon?“

■ Grafische Entwicklungsumgebungen schießen mittlerweile wie Pilze aus dem Boden. Im Dickicht der Angebotsfülle haben sich in den letzten zehn Jahren viele Werkzeuganbieter dazu verleiten lassen, die auf dem Markt angebotenen Tools zur grafischen Erstellung von Anwendungen auf zwei scheinbar unvereinbare Erscheinungsformen zu reduzieren: Der einen werden die rein grafischen Programmiersprachen zugeordnet, die hinsichtlich ihrer Flexibilität und Einsetzbarkeit mit den textorientierten Programmiersprachen vergleichbar sind, der anderen die rein blockorientierten Umgebungen, bei denen eine Gesamtapplikation durch Verknüpfung von Funktionsblöcken entsteht.



*RAHMAN JAMAL
ist Technical &
Marketing Director
National Instru-
ments, Central
Europe*

Diese überbetonte, bewusst herausgestellte Einteilung verfolgt den Zweck, einerseits die grafische Programmiersprache dem Anwender als zu komplex einzureden mit der unterschwelligen Botschaft „Programmiersprachen sind für Programmierer gedacht, nicht für Anwender“. Andererseits dient diese Kategorisierung als Mittel, grafische Konfigurationstools dem Anwender schmackhaft zu machen mit dem suggestiven Unterton

„Wer möchte schon programmieren?“. Vernehmbar ist da auch oft die pauschale Behauptung, man entferne sich einer Erscheinungsform umso mehr, je mehr man sich der anderen nähere. Bei nüchterner Betrachtung der Situation jedoch zeigt sich, dass sich streng genommen die Gruppe der Nicht-Programmierer wiederum in weitere Anwendertypen unterteilt. Die Spanne reicht dabei von Anwendern, die sich mit dem reinen Bedienen vordefinierter Applikationen begnügen, über solche mit wenig Erfahrung in der Programmierung, die eher das logische Verknüpfen konfigurierbarer Teillösungen zur Gesamtlösung bevorzugen, bis hin zu denjenigen, die im Grunde zum vorigen Typ zählen, aber in Ermangelung vorhandener Möglichkeiten gezwungen sind, auf textorientierte Skriptsprachen zurückzugreifen. Letzteres ist klassischerweise immer dann der Fall, wenn das Potenzial des eingesetzten grafischen Werkzeugs ausgeschöpft ist und die gewünschte Funktionalität über eine nicht konfigurierbare Methodik erzwungen werden muss. In der Regel erfordert ein solcher Konventionsbruch aber immer einen höheren Aufwand als die reine Bedienung des Konfigurationswerkzeugs.

Einen Gegensatz dazu bilden die grafischen Programmierer, die zwar die Flexibilität und Freiheit einer vollwertigen Programmiersprache suchen, sich aber nicht mit den syntaktischen

und semantischen Details einer konventionellen, textorientierten Programmiersprache auseinandersetzen, sondern sich eher ihren eigentlichen Aufgaben widmen möchten. Diese Klasse der Anwender verfügt meist über elementare Programmierkenntnisse, die aber dennoch in keinem Verhältnis zu dem speziellen Wissen etwa eines Informatikers stehen.

Augenblicklich löst sich die scheinbare Gegensätzlichkeit der Anforderungen und Positionierungen grafischer Methodiken auf, ermöglicht man nur den unterschiedlichen Anwenderklassen auch unterschiedliche Einstiegsmöglichkeiten in eine grafische Entwicklungsumgebung. Genau dieses Ziel verfolgt der Express-Gedanke in der aktuellen LabVIEW-Version. Erstmalig gelingt es durch die Express-Technologie in LabVIEW 7 Express, mit den Vorzügen der grafischen Applikationsentwicklung eine breite Anwenderschicht anzusprechen, wobei individueller Erfahrungs- und Kenntnisstand keine Rolle mehr spielen. Denn schließlich gilt das Interesse des Anwenders seiner ganz bestimmten Applikation und nicht der vom Hersteller vorgegebenen Klassifizierung der Werkzeuge. ■

Beitrag als PDF auf www.duv24.net

more @ click

TG0408

