Christian Tenhumberg

Flexibilität, Geschwindigkeit und Leistung

TestStand ist ein Managementsystem zur Entwicklung und Verwaltung von Testschritten. Herstellern aus der Elektronikindustrie wird damit ein Werkzeug zur Verfügung gestellt, das die Kosten bei der Entwicklung von Testsystemlösungen im Bereich Software minimieren hilft.

National Instruments blickt auf eine lange Tradition bei der Erstellung von Softwareentwicklungsumgebungen (LabVIEW, Measurement Studio, etc.) zurück. Nicht nur deshalb scheint man mit dem Produkt TestStand wohl relativ erfolgreich zu sein. Von der Zeitschrift Test & Measurement World (www.tmworld.com) wurde es sogar als „Best in Test"-Produkt 2002 ausgezeichnet. Bereits bei 85% der Top-Elektronikfirmen (Electronic Buisness Top 300 Electronic Companies 2000) sorgt es für effizientes Testen.

Standardisierte Oberfläche

Zwischen unterschiedlichen Testsystemen findet man viele Gemeinsamkeiten, angefangen bei den Softwaremodulen bis hin zur Hardware. Alle Softwarekomponenten, die für ein Testsystem ständig vorhanden sein müssen, sind deshalb bereits in TestStand integriert – von der Benutzerverwaltung bis zur Ablaufverfolgung. Typisch für eine Anwendung von TestStand ist z.B. die Prüfung eines Mobiltelefons. Hier werden die Prüfschritte für eine Fertigung erst am Ende des Entwicklungsprozesses definiert, was zusätzliche Kosten verursacht. Die Entwicklungsarbeit für einzelne Testschritte wurde aber eigentlich schon im Labor erledigt. Für den Produktionstest werden dann modifizierte Testschritte entwickelt, die oftmals in anderen Entwicklungsumgebungen erstellt werden. Hier bietet TestStand die Flexibilität, alle Abteilungen, die in den Entwicklungsprozess eines Produkts eingebunden sind, einzubeziehen. Dazu gehören natürlich auch die Bereiche Wartung und Support. Die Integration in das neue Testkonzept beschränkt sich also auf die besonderen Eigenschaften des Testsystems – mit der Software als Werkzeug ist man ja bereits vertraut.
Die Schulungen der Mitarbeiter auf das neue Testsystem können hierbei fast entfallen. Dagegen wächst der gemeinsame Kenntnisstand zwischen den einzelnen Abteilungen dadurch, dass auf eine Testplattform und eine optimierte Teststrategie zugegriffen werden kann. Dies schlägt sich erheblich auf die Reduzierung der Kosten und der Time-to-Market nieder.

Bild 1: Unterschiedliche Schritttypen des DLL-Adapters

Kompatibilität

Ein weiterer wichtiger Punkt ist die Kompatibilität der derzeitigen Hardware mit dem jeweiligen Betriebssystem und den folgenden Versionen, wie zurzeit aktuell Windows XP. Oftmals ist es sehr arbeits-, zeit- und kostenintensiv, eine Portierung auf ein neues Betriebssystem vornehmen zu müssen. National Instruments hat sich bemüht, den Aufwand der Portierung von TestStand so gering wie möglich zu halten. Als Standard dient in diesem Fall die Anwendungssoftware und nicht das jeweilige Betriebssystem.
Durch die unterschiedlichen Schnittstellen von TestStand, wie z. B. IVI (Interchangeable Virtual Instruments) erhält man die Flexibilität, Produkte von unterschiedlichen Hardwareherstellern einzusetzen. Derzeit existieren IVI-Treiber für über 3 000 verschiedene Modelle unterschiedlicher Hersteller (www.ni.com/devzone/ident).

Schnittstellen

Wie oft wurden schon Softwaremodule entwickelt, die eigentlich schon existieren? Diese mussten oftmals nur deshalb neu entwickelt werden, weil beispielsweise die Schnittstelle nicht mehr zum neuen Testsystem passte. Die angedachte Modularität wurde nicht streng genug umgesetzt, so dass einige Softwaremodule ein gewisses Eigenleben führen. Deshalb musste man oftmals das Testmanagementsystem dem Softwaremodul anpassen. Derartige Änderungen bedeuten aber manchmal einen unvorstellbaren Aufwand. Mit TestStand hingegen erhält man Schnittstellen für die jeweiligen Softwaremodule, durch die man gezwungen wird, sich an Regeln zu halten – ein wichtiger Punkt bei der Modularisierbarkeit und Wiederverwendbarkeit von Softwaremodulen. Diese Schnittstellen bzw. Softwareadapter sind

- DLL Flexible Prototype Adapter,
- LabWindows Standard CVI Prototype Adapter,
- LabVIEW Standard Prototype Adapter,
- Automation Adapter (ActiveX Automation Server),
- HTBasic Adapter,
- Sequence Adapter (Aufruf von Untersequenzen) sowie
- <None> Adapter (Schritt ohne Softwaremodul).

Mit dem DLL Flexible Prototype Adapter kann man Funktionen aus DLLs (Dynamic Link Libraries) aufrufen, die in den unterschiedlichen Entwicklungsumgebungen (Borland Builder C++, Microsoft Visual C++, etc.) entstanden sind. Hierbei sind die Standards der Entwicklungsumgebungen für das Struct Packing unterschiedlich. Microsoft Visual C++ und Symantec C++ benutzen standardmäßig 8-Byte-Packing. ▷

Bild 2: Vergleich der Freitagsversion mit derjenigen vom Montag

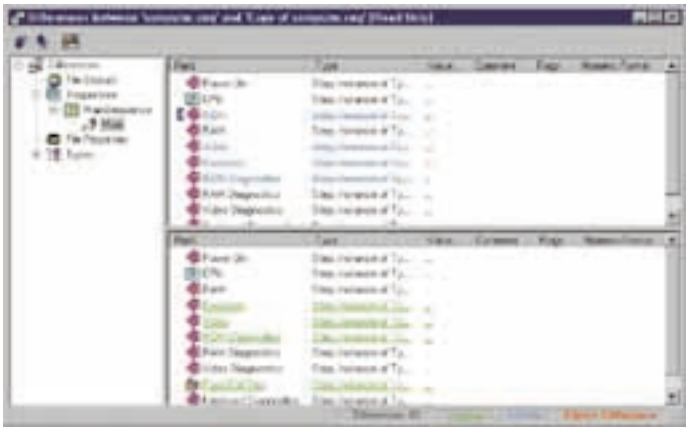

Bild 3: Mit Hilfe der Synchronisierungsschritte können Daten zwischen mehreren Threads bzw. Prozessen oder Operationen ausgeführt werden

Im Gegensatz dazu verwenden LabVIEW, Borland C++ und Watcom C++ 1-Byte-Packing. Aber auch dies ist in TestStand mit wenigen Mausklicks konfigurierbar, falls es nicht in der jeweiligen DLL programmiert worden ist (z. B. #pragma pack(1)). Weiterhin werden Code Templates, sowie unterschiedliche Schritttypen, standardmäßig dem jeweiligen Adapter hinterlegt, so dass man sich direkt eingehend mit der Messaufgabe beschäftigen kann.

Bild 1 zeigt die unterschiedlichen Schritttypen des DLL-Adapters, die standardmäßig vorgegeben sind. Für jeden anderen Adapter erhält man eine ähnliche Auswahl an Schritttypen. Alle Schritttypen der jeweiligen Adapter kann man auf einfache Weise modifizieren, aber ein Großteil der in der Praxis geforderten Schritttypen wird mit dieser Auswahl bereits abgedeckt. Ferner erkennt man die Datenbank-, Synchronisierungs- und IVI-Schritte, die in einem späteren Abschnitt erläutert werden.

Während des Entwickelns der Softwaremodule muss der Quellcode getestet werden, wobei die Befehlszeilen oftmals Schritt für Schritt debuggt werden. Hier stellt sich nun die Frage, ob man zum Debuggen die Entwicklungsumgebung wechseln muss. Dies ist nicht der Fall, denn wenn der Adapter vorhanden ist, wird mittels ActiveX die jeweilige Entwicklungsumgebung geöffnet. Beispielsweise hat man ein Softwaremodul in LabVIEW geschrieben und möchte dieses nun testen. Hierzu muss man kein zusätzliches Softwareprogramm schreiben, sondern kann das LabVIEW-Modul von TestStand aus ausführen.

Ein weiteres Beispiel wäre eine DLL, die man mit LabWindows/CVI entwickelt hat. Hierbei kann man von TestStand aus den Quellcode direkt in LabWindows/CVI debuggen. Sollten diese Punkte noch nicht flexibel genug sein, besteht noch die Möglichkeit, eigene Adapter oder eigene Schritttypen zu entwickeln. TestStand ist flexibel, und man kann es an die eigenen Bedürfnisse anpassen, wenn die vorgegebenen Standards nicht allen Änderungen gerecht werden.

Die erstellten Codemodule müssen in irgendeiner Form verwaltet werden, d. h. man muss wissen, welche Module bereits existieren und in welchen Versionen diese vorliegen. In die Version TestStand 2.0 sind Projektverwaltungswerkzeuge integriert worden, die eine Aufteilung in Arbeitsbereiche und dazugehörige Projekte ermöglichen. Hierbei erhält man eine Art Explorerübersicht zu allen Projekten, Codemodulen, Sequenzen und sonstigen Dateien, die in den jeweiligen Projekten benötigt werden. Zu jeder Datei erfährt man den zugehörigen Projektnamen, die Versionsnummer und ob sie in SCC (Source Code Control) eingebunden ist. Diese Schnittstelle SCC ist der derzeitige Standard.

In Verbindung mit Microsoft Visual Source Safe lassen sich z. B. große Projekte, an denen mehrere Entwickler gleichzeitig arbeiten, verwalten. Mit diesem Werkzeug werden, vereinfacht gesagt, Codemodule ein-

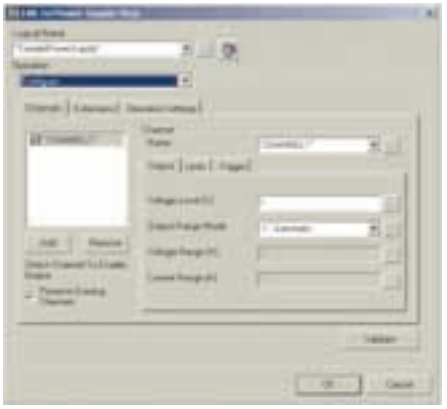

Bild 4: TestStand-Schritttyp für die Klasse der Netzteile

bzw. ausgecheckt und sichergestellt, dass jeder Entwickler die aktuellste Version des Codes hat. Damit Änderungen zwischen den verschiedenen Versionen (ohne SCC) übersichtlich erkannt werden können, bietet TestStand ein Vergleichswerkzeug. Hierbei werden alle Unterschiede der beiden Sequenzen übersichtlich dargestellt. Ein praktisches Beispiel wäre etwa der Vergleich der Freitagsversion mit derjenigen vom Montag (**Bild 2**).

Darstellung der Eregbnisse

Ein weiterer wichtiger Punkt bei der Einsparung der Entwicklungszeit ist die Darstellung der Ergebnisse. Hierbei kann direkt ausgewählt werden, welche Art von Bericht erstellt werden soll (ASCII, HTML, XML). Der Standardbericht muss lediglich konfiguriert werden, d. h., man gibt an, ob Einzelschrittergebnisse in den Bericht eingebunden werden sollen, wann dies der Fall sein soll (z. B. nur wenn die Einzelschritte bestanden worden sind), in welchem Verzeichnis diese Ergebnisse abgelegt werden sollen usw.

Da die Berichterstellung wiederum in Sequenzen untergebracht ist, die mit den Schritttypen von TestStand erstellt worden sind, können auch hier Änderungen am Aussehen bzw. Format vorgenommen werden.

Ebenso wie die Berichterstellung kann die Datenbankanbindung konfiguriert werden. Im Betriebssystem Windows gibt es einige Schnittstellentechnologien für die Anbindung an Datenbanken. TestStand verwendet die Schnittstelle ActiveX Data Objects (ADO) als Hauptdatenbanktechnologie. Unterhalb von ADO liegt OLE DB (Object Linking and Embedding Database). Hierbei nutzt TestStand ebenso wie alle anderen Programme, die ADO verwenden, indirekt OLE DB. Der OLE-DB-Schnittstellenlayer bzw. ODBC (Open Database Connectivity) schafft nun direkt eine Verbindung durch den OLE DB Provider zum DBMS (Data Base Management System). Typische OLE DB Provider sind Access, Oracle, SQL-Server. Hierbei stellen dann die OLE DB Provider die Verbindung zu den eigentlichen Datenbankdateien her.

Mit TestStand muss man nicht sehr tief ins Detail gehen, um eine Datenbankverbindung aufzubauen. Hierzu reicht der Data Link, der den Pfad und den Namen der Datenbankdatei beinhaltet, sowie die Konfiguration der Datenbankoptionen, welche Ergebnisse in welche Spalten eingetragen werden sollen. Falls noch keine Datenbank existiert, kann diese mit dem jeweiligen Treiber (Systemsteuerung > ODBC Data Sources) erstellt werden. Die so erstellte Datenbank besitzt aber noch keine Struktur bzw. Tabelle. Diese kann entweder mit ▷

Hilfe des jeweiligen Anbieters oder auch mit TestStand erstellt werden. Hierzu ist in TestStand das Werkzeug „Database Viewer“ integriert worden.
Mit dem Database Viewer kann man SQL-Befehle ausführen oder einfach nur die Dateien, die sich bereits in der Datenbank befinden, ansehen. Durch Laden der Datenbanktemplates von TestStand können einer Datenbank Strukturen verliehen werden. Hierzu sind Vorgaben für Oracle, Access und SQL vorhanden.
Zudem ermöglicht die Version TestStand 2.0 nun Multithreading. Mit Hilfe der Synchronisierungschritte können Daten zwischen mehreren Threads bzw. Prozessen oder Operationen ausgeführt werden (**Bild 3**). Für diese Schritte ist keine Programmierung, sondern nur eine Konfiguration erforderlich.
Will man einen exklusiven Zugriff auf eine Resource gewährleisten, so benötigt man den Schritt „Lock“. Eine Resource kann ein Softwaremodul, eine Datenbankverbindung, eine Datenerfassungkarte oder ähnliches sein. Mit Semaphoren hingegen limitiert man den Zugriff auf eine bestimmte Anzahl von Threads auf eine Resource. Setzt man bei dem Schritt „Semaphore“ die Anzahl der Threads auf 1, so hat man einen Lock-Schritt nachgestellt. Rendezvous-Schritte wiederum verwendet man z. B. dafür, um Ergebnisse parallel arbeitender Sequenzen in eine Datenbank zu schreiben. Dies soll nur ein einfaches Beispiel sein – normalerweise würde man die Datenbankverbindung vielmehr als Hintergrundprozess arbeiten lassen. Queue-Schritte werden als eine Informationsschnittstelle zwischen verschiedenen Threads genutzt. In dieser Queue können Daten abgelegt oder gelesen werden. Ein weiterer Schritttyp sind die Notifications, mit denen ein oder mehrere Threads darüber informiert werden, dass ein bestimmtes Verhalten oder ein besonderes Ereignis aufgetreten ist. Der Schritt „Batch“ findet Verwendung bei gleichen Prüflingen, die simultan getestet werden, wie bei einem MultiHeadTester.

IVI und praxisgerechte Treiber

Im Jahr 1998 formierten sich National Instruments und andere Firmen zur Interchangeable Virtual Instruments Foundation (IVI Foundation). Diese wurde gegründet, um Standards für Gerätetreiber im Bereich Design und Adressierung zu entwickeln und zu standardisieren. Weiterhin sollten diese Treiber in der Lage sein, einen Simulationsmodus zu unterstützen und redundante Befehle zu eliminieren. Das Hauptmerkmal besteht in der Austauschbarkeit der Geräte, ohne dass der dem Gerät zugrundeliegende Quellcode verändert werden müsste. Bisher wurden fünf Geräteklassen festgelegt:

- Digitalmultimeter,
- Oszilloskope/Digitizer,
- Signalgeneratoren,
- Schaltgeräte sowie
- Netzteile.

Die Treiber sind praxisgerecht und ermöglichen eine Minimierung der Gerätekosten z. B. im Bereich Kalibrierung. Ein Gerät sollte also unabhängig vom Hersteller oder der Schnittstelle (GPIB, VXI, etc.) austauschbar sein. TestStand besitzt bereits die o. g. Schritttypen für alle fünf Klassen. Somit muss mit diesen Schritttypen nur noch die gewünschte Funktion ausgewählt werden. Ebenso lassen sich mit diesen Schritten Geräte simulieren und ihre Entwicklung fortsetzen. Dies gilt auch für den Treiber. **Bild 4** zeigt den TestStand-Schritttyp für die Klasse der Netzteile (Power Supply).

Fax 0 89/7 14 60 35
www.ni.com/teststand
propductronic **404**

Dipl.-Ing. (FH) Christian Tenhumberg *ist Applikationsingenieur bei der National Instruments Germany GmbH in München.*