

## **Delivering On Obsolescence Protection**

White Paper  
by Hob Wubbena, Agilent Technologies

© 2007 IEEE. Reprinted, with permission, from  
IEEE publication title: #1569048925

# Delivering On Obsolescence Protection

**Abstract** – With the growing need to reduce operational costs, many aerospace and defense companies are focusing on lowering the overall lifecycle costs of test platforms. The difficulty of this challenge is often compounded by the need to sustain test systems over the multi-decade life of the supported platform—a requirement that typically increases the long-term maintenance costs of an aging automated test system (ATS).

In many cases, the shortcomings of existing test system architectures contribute to the problem. An alternative approach, already in use, addresses those shortcomings through an efficient, interactive platform for creating flexible system frameworks. This platform simplifies the process of replacing discontinued instruments with either similar devices or alternatives such as modular instruments or abstract instruments.

Most test-system architectures typically require standardized hardware, software, instrument interfaces or communication protocols. In contrast, the “managed integration” alternative uses a services-based approach to enable inclusion of wide-ranging languages, drivers, interfaces, protocols, test executives and instrument types into hybrid systems that easily combine new and existing elements.

To mitigate instrument or software obsolescence, this services-based platform completely decouples the hardware and software. This allows component connections to be configured (e.g., created and deleted) at runtime via “interaction services” without compilation, linking, or pausing of the system. Thus, when an instrument must be replaced, it can be hot-swapped by changing the connections even if its predecessor used a different communication protocol.

# Introduction

As the military continues to shift funds from programs to operations, program developers are putting greater emphasis on efficiency, taking such steps as the consolidation of test platforms and the migration of existing test systems. Further, to maximize funding within the operational budget for troops and their direct support, the operational investment in test systems is being made more effective by extending the life of existing hardware and protecting against obsolescence (e.g., eCASS, AVITS).

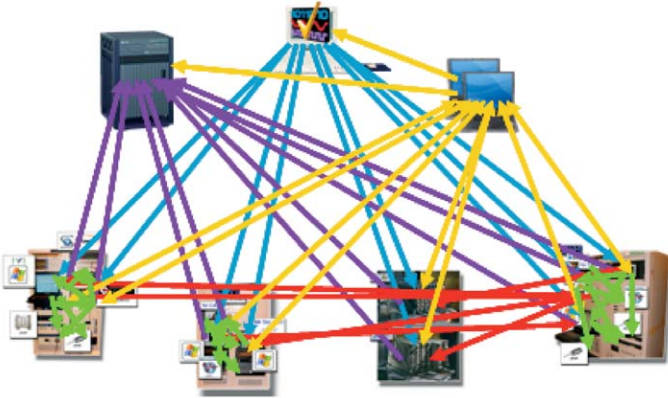
One of the most effective ways to reduce operational costs is to lower the lifecycle costs of automated test systems (ATS). Various initiatives have identified key opportunities for improvement:

- Total cost of ownership (TCO) for an ATS
- The time and effort required to develop and deploy an ATS
- Migration and reuse of test program sets (TPS)
- The physical footprint of an ATS
- The logistics footprint of an ATS (e.g., spares, training)
- Interoperability across U.S. and allied services

The need for these improvements is amplified by the need to sustain an ATS over the 20- to 30-year life of the supported platform. This requirement is in stark contrast to the product lifecycles in commercial industries such as information technology (IT), software and test equipment. As these industries become more intensely competitive, their respective product lifecycles become shorter and shorter.

These are certainly difficult challenges. Unfortunately, they are often made more difficult by typical approaches to test-system architecture and integration. For example, current integration methods require standard interface protocols (e.g., IVI for instrument drivers or DLLs/COM. NET for software) that directly couple hardware devices to the software. This architectural approach requires moderate to intense levels of customization to completely fulfill a set of test requirements.

The root of system complexity begins with the typical star-shaped connection of system elements (Figure 1).



**Figure 1. The complexity of traditional test systems begins with the star pattern of interconnection.**

This complexity is exacerbated by the typical sequence used to integrate and automate a system:

1. Identify system components, protocols and rack distribution
2. Identify, negotiate and document connectivity for both control and data
  - a. For P protocols, implement up to  $P^2$  protocol conversions
  - b. For C devices or elements, implement up to  $C^2$  connections
3. Integrate using multiple development tools and diverse protocol expertise (for each connection)
4. Design, implement, test and validate each leg of the star
5. Interview one or more platform experts before programmatically automating the ATS
6. Hope no additional system requirements are imposed during the development process (often an unrealistic hope in practice)

To meet the full scale and scope of the test requirements, it is often necessary to mix a variety of instrument types. For example, a modular approach (e.g., PXI cardcage or LXI-based instrument) may be the best way to implement system switching. A variety of rack-mountable bench-type instruments may be required to provide the necessary radio-frequency (RF) or microwave measurement capabilities.

Each piece of system hardware may have a different option for I/O: GPIB, LAN, USB, RS-232C, Firewire and so on. System complexity rises along with an increase in the number and type of I/O links that must be used.

For the test-system software, practicality often dictates that the ATS use just one application or language, based on the experience of the software developer and the number of previously written programs available to the team. While this may simplify the programming task, it may also lead to compromises in functionality, performance and ease of use.

There will also be a variety of communication protocols associated with the instruments, I/O and software used in the system. Drivers simplify instrument replacement; however, because drivers usually access only a subset of an instrument's full functionality it is often necessary to create instrument-specific code to use broader functionality. What's more, Interchangeable Virtual Instrument (IVI) drivers include eight separate protocols (e.g., IVI-Scope, IVI-Function Generator). If any of these are customized beyond the class driver—and this is common because most class drivers cover about half of an instrument's functionality—then each customization within each class becomes a new protocol. All of this only serves to increase system complexity, lengthen development time and make system testing more difficult.

As a star-structured ATS ages, it will become increasingly difficult to migrate from old to new instrumentation, I/O, software or communications protocols. Software will often be the biggest problem: Typically, less than 20 percent of all test-routine code that controls the interfaces between devices and software can be reused, only 20 percent can be modified, and the remaining 60 percent must be written from scratch. [1] Consequently, changing just one instrument—upgraded, out for service or discontinued—can require costly, time-consuming software changes. Depending on the extent of the changes, costly and time-consuming re-qualification of the ATS may also be required.

## Overcoming the Challenges

An alternative architectural approach provides an efficient, interactive platform for creating flexible system frameworks. Currently available, this “virtual rack” (VR) platform reduces the time and effort required to integrate, automate, maintain and evolve test systems. It does this in part by completely decoupling instrumentation, I/O and software, isolating each element within the VR platform. [2]

To illustrate the difference, consider system creation with a typical test executive. Suppose a specific power-up sequence (as defined by the device designer) is tied to the voltage reading from a digital multimeter (DMM) at a specific time. The software-based test, the DMM and the I/O to the DMM are all tightly linked—and a change in any one will likely require changes to the other two. By decoupling all three elements, the power-up sequence can be created separately from the communications to the DMM and the specific I/O link.

With the VR platform, integration is independent of specific interfaces or programming languages. The key concept is “managed integration,” which means the desired state of integration is not programmed but instead configured through an interaction service. The system architect or test engineer must specify only the endpoints—hardware, firmware, software components—and the VR platform handles all of the required communications and translations. This is enabled by interaction sources and interaction sinks without compilation, linking, or pausing of the system.

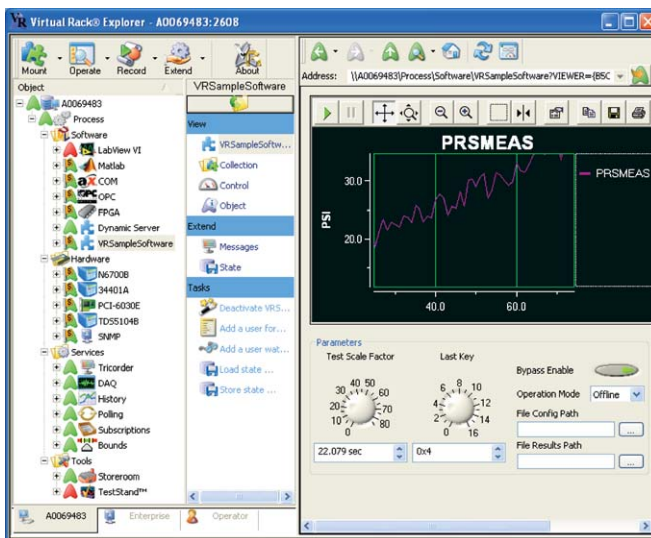


Figure 2. The VR explorer is used to design, manage and operate a measurement system.

## Applying the System Platform

A test system that uses the managed-integration approach can be created by following the steps spelled out in the acronym **MORE: mount, operate, record and extend**.

**Mount:** Components are mounted into the VR platform using a mouse to drag and drop elements housed in the VR “storeroom.” The elements can be hardware devices, test equipment, benchtop instruments, software modules, applications, firmware and VR services. System creators can also add custom hardware, software or firmware. Any VR configuration can be saved and recalled. [3]

**Operate:** This can begin immediately using the VR Explorer (Figure 2), which works like a combination of a file explorer (e.g., tree view) and a Web browser. The VR explorer is also used to design and manage measurement systems. [4]

**Record:** The VR macro recorder is used to create automated sequences and tests, including a complete series of system actions. Macros can be created by simply operating the equipment and the captured sequence of actions can be revised with the sequence editor. [5]

**Extend:** New capabilities can be added to the system with drag-and-drop simplicity—and without modifying the original components, keeping them pristine from project to project. Built-in services include state with caching, bounds (limits), history, archiving, mnemonics, telemetry and many others. [6]

Thousands of hardware and software components are already included in the VR storeroom and new components can be added using a developer’s kit. A component contained in the storeroom is made accessible via a Virtual Rack Mount Kit (VRMK) and can be thought of as an “integration driver,” which typically takes just a few days to create rather than the weeks or months it might require to develop an instrument driver. The VRMK can also use any driver or just standard instrument commands (e.g., SCPI tree, etc.) to integrate the device. In contrast to an instrument driver, a VRMK can also dynamically or statically integrate firmware as well as software such as C code, a MATLAB® program, a LabVIEW vi, a TestStand application or an Agilent VEE Pro program.

This approach supports the need to leverage existing assets into a new or modified ATS, or to replace instruments that are out for service or have been discontinued. In addition to enabling a gradual transition from present to future architectures, it also supports the interim need to create and use hybrid systems that utilize all types of instrumentation (e.g., GPIB, VXI, PXI, LXI), interfaces (e.g., GPIB, Firewire, LAN, USB) and software (e.g., Agilent VEE Pro, LabVIEW, ATEasy).

## Exploring a Typical Scenario

To illustrate the advantages of a managed-integration platform in the context of an ATS, let's consider a typical scenario: replace an Agilent E3631A triple-output power supply with a modular Agilent N6700B that has four outputs. The existing supply has a GPIB interface while the new one has both GPIB and LXI/LAN interfaces.

Three key elements of the managed-integration platform enable efficient instrument replacement. First is the VR storeroom, mentioned earlier. The thousands of existing storeroom components include a wide range of hardware, software, firmware, I/O and protocols.

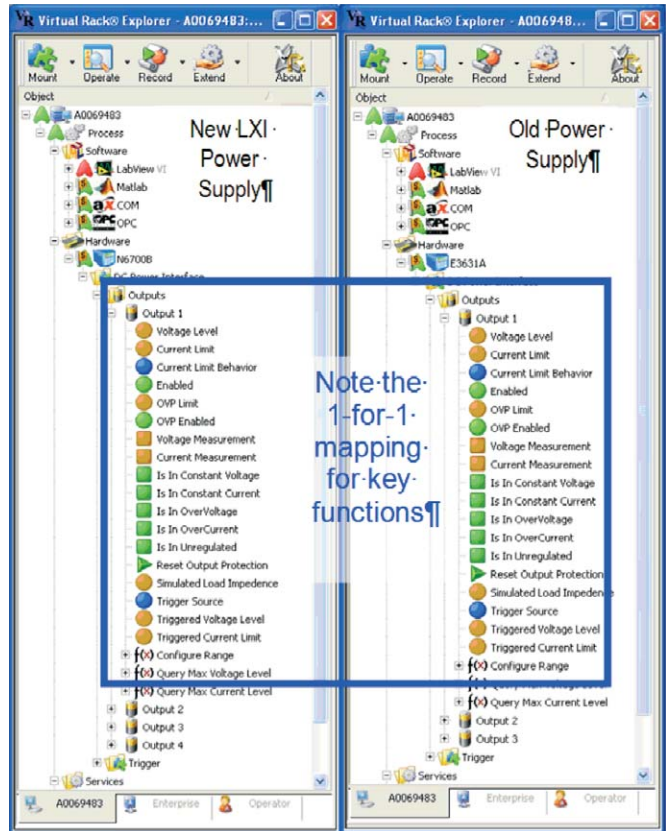
The second is the inherent simplicity of the managed integration approach compared to integration based on programming languages or standard interfaces. Languages and interfaces typically have hundreds of commands that represent the functionality of the application or device as well as the I/O bus.

As an example, let's look at the replacement scenario. The function call for the E3631A is from a *VXIplug&play* driver (commonly used with this supply) sent via GPIB. The equivalent function call for the N6700B uses an IVI-COM driver (Table 1). The commands are completely different, and are specific to the instrument and its underlying bus address.

**Table 1. The old and new power supplies require different function calls that depend on the type of driver being used.**

Device	"Set Voltage" Commands
E3631A	<code>Hpe3631a_voltage(instrHandle, hpe3631a_VOLT_INST1, "")</code>
N6700B	<code>voltageLevel = ilviDCPwr.Outputs.Item("").VoltageLevel</code>




In contrast, VR-based managed integration uses just three "verbs" or actions: *Set*, *Get* and *Do* for all software, hardware and firmware (Figure 3). All functions are mapped to these actions, and their specifics become parameters. From this, we can say integration is enabled by very few verbs (just three) and many nouns (hundreds). This is the opposite of the approach used by languages and interfaces, which rely on hundreds of verbs (actions) and very few nouns (parameters).



**Figure 3. VR's Set (circle), Get (square) and Do (triangle) enable easy mapping during instrument or software replacement.**

As illustrated in Table 2, VR uses the same commands for key functions (e.g., set voltage level, check for constant-voltage state, reset output protection) even though the E3631A is GPIB-based and the N6700B is an LXI-compliant device that can be accessed via LAN. These managed-integration commands are also independent of whether the instrument is accessed from a driver, through an interface, or directly using standard instrument commands.

**Table 2. VR uses the same commands for either power supply.**

Command	E3631A or N6700B
<b>Set</b>	 Voltage level
<b>Get</b>	 Is in constant voltage
<b>Do</b>	 Reset output protection

The third key element of managed integration is the addition of services and behaviors to each component. These are independent of each component but the interaction service makes them available to every component. Specific VR services include macro recorder, limit checking, state, history, polling and interaction. Interaction is one of the key VR services: it enables the addition of specific behaviors to any component through sourcing and sinking.

This “delegated services” approach adds unique component behavior to a test system while isolating any system modifications from the original components. There is no need to modify or retest components due to system modifications because the components are identical to the original components. Figure 4 illustrates an NI-DAQ system with bounds checking, history and interaction sets.

Because the VR interaction service is a behavioral model, the interaction relationship is established at the lowest level of granularity possible (an interactor) with the instrument: each property, method or event is accessed through whatever interface is provided (e.g., a SCPI function call). Once the new power supply is physically added to the system, the new property is dragged into the interaction spreadsheet without any reprogramming.

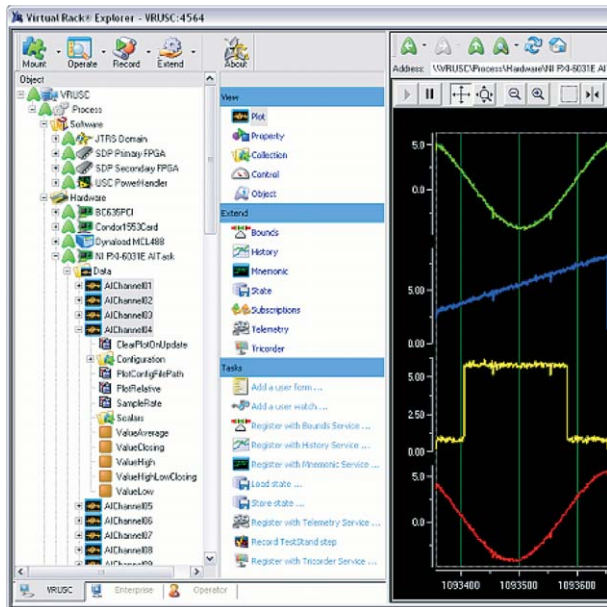


Figure 4: The application of configurable services to a test system enables the addition of new behaviors to system components.

Creating an interaction for the replacement instrument is a simple two-step process within the VR framework. The first step is finding the property “power supply (source)” in the VR explorer tree and then dragging the interaction source into an interaction spreadsheet. The second step

is to find the interaction sink channel/attribute of the “voltmeter (sink)” in the VR explorer tree and drag it into the interaction-sink cell of the spreadsheet. VR then verifies that the relationship is valid (Figure 5).

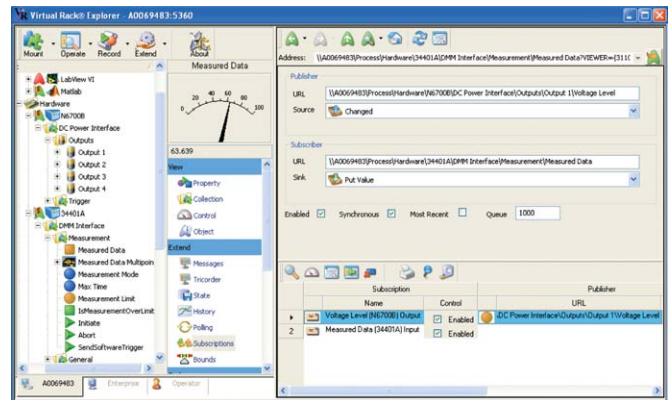


Figure 5: VR verifies both the publish and subscribe relationships of a replacement instrument.

This simple drag-and-drop approach lets the system architect or test engineer focus on *what* is connected without spending time on the *how*—and the resulting functionality is typically equivalent to entire sections of code.

However, when swapping a new component in for an old one, this approach requires a search-and-replace operation to update every obsolete publication and subscription (this can be tedious but is relative easy to implement). Also, this method covers neither the “don’t cares” of the new component nor the component-specific initialization that changes when a new component is added.

To overcome this, longevity can be designed into a system by creating an intermediate or abstract layer for each component. This can be done quite easily using the method described above, but inserting an abstract component (rather than a real component) such as an abstract power supply and using this abstract component in all recordings, GUIs, histories and so on. This is an essential piece of the solution because all tests, GUIs and services will reference the abstract, which never changes. The abstract component represents the key to automatic obsolescence protection for a specific component within the system. This process enables the swapping of components (hardware, software and firmware) using a configuration instead of programming.

This approach also enables the replacement of one or more hardware or software components, no matter which communication protocols they use. The process is identical even if the old power supply was connected via GPIB and the new supply is connected via LAN.

## Conclusion

The VR managed-integration approach provides an efficient, interactive platform for creating flexible system frameworks. This approach also avoids a crucial shortcoming of traditional star-connected test system architectures: with managed integration, a change in one element no longer affects every connected element. By completely decoupling the major system elements and enabling simple interactions, the VR platform reduces the time and effort required to integrate, automate, maintain and evolve test systems. Once all system elements have been added to the VR storeroom and converted into configurable services, replacement of any system element—hardware, I/O, software, protocol—becomes drag-and-drop easy.

Any new system created with the VR platform can help military services, prime contractors and subcontractors extend the life of existing test assets. These can be reused in hybrid systems that utilize best-in-class system elements that address ATS requirements. Once such a system has been deployed, the VR platform protects the ATS against obsolescence by greatly simplifying the process of replacing an instrument, a software application, an I/O connection or a communication protocol. The resulting reduction in effort is clearly advantageous to those seeking to reduce operational costs while still ensuring high quality testing of long-lived platforms.

## References

- [1] From survey data prepared by Griggs-Anderson Research for Agilent Technologies, c.2006
- [2] Agilent VR brochure, 2007
- [3] Agilent VR Quick Start Guide, 2007, pp 11-18
- [4] Agilent VR Quick Start Guide, 2007, pp 18-29
- [5] Agilent VR Quick Start Guide, 2007, pp 29-35
- [6] Agilent VR Quick Start Guide, 2007, pp 35-41
- [7] Agilent VR Subscription Service Manual, 2007, pp 3-10

---

MATLAB is a U.S. registered trademark of The MathWorks, Inc.

## Remove all doubt

Our repair and calibration services will get your equipment back to you, performing like new, when promised. You will get full value out of your Agilent equipment throughout its lifetime. Your equipment will be serviced by Agilent-trained technicians using the latest factory calibration procedures, automated repair diagnostics and genuine parts. You will always have the utmost confidence in your measurements.

Agilent offers a wide range of additional expert test and measurement services for your equipment, including initial start-up assistance onsite education and training, as well as design, system integration, and project management.

For more information on repair and calibration services, go to:

[www.agilent.com/find/removealldoubt](http://www.agilent.com/find/removealldoubt)



[www.agilent.com/find/agilentdirect](http://www.agilent.com/find/agilentdirect)

Quickly choose and use your test equipment solutions with confidence.

## Schedule a Live Demo

Contact us to schedule a live demo of how Virtual Rack can integrate your system in just minutes and let you start recording procedures immediately.

For more information about Virtual Rack, visit:  
<http://www.agilent.com/find/virtualrack>

For Virtual Rack email assistance:  
[virtualrack@agilent.com](mailto:virtualrack@agilent.com)

For direct Virtual Rack phone assistance:  
**United States: (tel) 970 679 3889**

## [www.agilent.com](http://www.agilent.com)

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

## [www.agilent.com/find/contactus](http://www.agilent.com/find/contactus)

### Americas

Canada	(877) 894-4414
Latin America	305 269 7500
United States	(800) 829-4444

### Asia Pacific

Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 112 929
Japan	81 426 56 7832
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Thailand	1 800 226 008

### Europe

Austria	0820 87 44 11
Belgium	32 (0) 2 404 93 40
Denmark	45 70 13 15 15
Finland	358 (0) 10 855 2100
France	0825 010 700
Germany	01805 24 6333*
	*0.14€/minute
Ireland	1890 924 204
Italy	39 02 92 60 8 484
Netherlands	31 (0) 20 547 2111
Spain	34 (91) 631 3300
Sweden	0200-88 22 55
Switzerland (French)	41 (21) 8113811(Option 2)
Switzerland (German)	0800 80 53 53 (Option 1)
United Kingdom	44 (0) 118 9276201

Other European Countries:

[www.agilent.com/find/contactus](http://www.agilent.com/find/contactus)

Revised: May 7, 2007

Product specifications and descriptions in this document subject to change without notice.

© 2007 IEEE. Reprinted, with permission, from IEEE publication title: #1569048925

© Agilent Technologies, Inc. 2007  
Printed in USA, September 19, 2007  
5989-7286EN



Agilent Technologies