PUSHING THE LIMITS OF LABVIEW

Out Of This World: Testing Satellite Equipments with LabVIEW and VI Package Manager

By Chris Roebuck, Software Manager for Test and Validation, Payload Products Division, Astrium Ltd.
Certified LabVIEW™ Architect and Certified TestStand™ Architect

All the Space You Need

My company, Astrium Ltd., is Europe's leader in space transportation, satellite systems, and related services. Our company motto says it best: "Astrium – All the space you need."

Hot Bird 7. © Astrium.

Our UK-based team is responsible for designing, manufacturing and testing satellite payload equipments, typically radio frequency (RF) equipment such as frequency converters, power amplifiers, and digital signal processing (DSP) hardware. Our satellites handle everything from Earth observation to space exploration and telecommunications infrastructure. It's critical that our products provide 100% reliable in-orbit service, because there are no "field returns" from space – if it breaks, it stays broken and it may compromise the whole mission! That means our solutions have to be perfect.

Not surprisingly, our test campaigns are extremely comprehensive. We test everything from single hybrid subassemblies (the building blocks) to complete functional systems, such as multi-channel frequency converters and solid state power amplifiers. We run tests of all kinds, from physical tests such as shock and vibration to system performance test in a thermal vacuum. Some of these test campaigns are months in duration because quality is such a huge priority.

We use LabVIEW and TestStand extensively in our testing – together, they form an incredibly useful toolset to help our products approach 100% reliability/availability.

The Grail: A Shared VI Reuse Library

When I started working with Astrium Ltd., our team consisted of more than 30 people, a mix of contractors and employees. While many team members had a lot of LabVIEW experience, they didn't realize how much they could share and collaborate to make everyone more efficient. We

THE GOALS

- Eliminate duplication of work across a large development team
- Concurrently support multiple versions of LabVIEW
- Be able to instantly recreate any LabVIEW project environment

THE RESULTS

- Central repository of shared, reusable VIs with named owners
- More efficient development
- Easy "rollback" to previous project environments (coming soon)

THE ENVIRONMENT

- NI LabVIEW 8.2, 8.5, 8.6, and 2009
- NI TestStand 3.5 and 4.0.2
- Windows XP
- A wide range of hardware

ASTRIUM
AN EADS COMPANY

Anchorage Road
Portsmouth
Hampshire
PO3 5PU
UK
+44 (0) 23 9270 5705
www.astrium.eads.net

didn't have a good mechanism to share and receive updates on code, and of course nobody really wants to break anyone else's code, so when in doubt of status or ownership, we'd create something new.

As a result, we had a code base of tens of thousands of VIs with lots of duplication and no clear ownership of the code. Different team members would write their own versions of a VI, only to find others had written – and perhaps applied various patches to – other versions of the same thing. Not only was this process very inefficient as we duplicated effort, it had effects on code stability and quality too. We never knew which versions of which VIs had been thoroughly tested and proven.

We knew we needed a VI reuse library so that team members could own various parts of the code, test them thoroughly, and make them available to everyone. At a previous company, I had used JKI's VI Package Manager (VIPM), which is designed to help companies package and share LabVIEW code. So I knew it had several core features that could really help us:

1. **It supports multiple LabVIEW versions.** Because our systems demand such high reliability, platform stability is a paramount requirement. Astrium requires that we start each LabVIEW project on a particular LabVIEW version and stick with it regardless of how many upgrades NI releases. We currently have four versions of LabVIEW in use, with many developers working on projects across multiple versions. These developers need to be able to easily install all their required drivers safely and reliably in every version of LabVIEW they use.

2. **You can receive updates from VIPM's package repository to learn about new packages immediately.** Before we started using VIPM, our developers were working independently with no way to share code. This lack of process meant that different developers would often fix the same problem, each in their own way – and we don't have time to duplicate effort. With VIPM, when someone finds a problem, he or she can fix it and push it out to everyone instantly, so we all have the most updated and stable code.

3. **VI Package Configurations (VIPCs), a feature of VIPM Professional, let you recreate a test system as it existed at any point in time.** This feature is particularly important to us because our products have a long support life. By using a VIPC file for each project, we ensure traceability of our reuse code and can replicate the exact development environment of any project even years later.

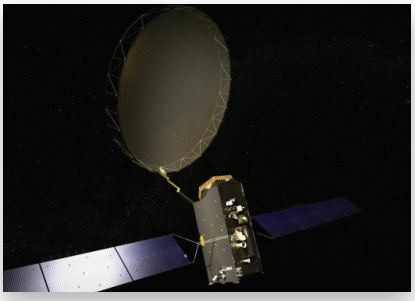
More Efficient Team, More Stable Code with VIPM

We have a vision of a huge reuse library with everyone pulling tools off the palettes. VIPM makes that vision possible.

Using VIPM Enterprise, we were able to create a networked VI Package Repository that allows individuals to own specific code modules. After

"We have a vision of a huge reuse library with everyone pulling tools off the palettes. VIPM makes that vision possible."

– Chris Roebuck



Alphasat I-XL. © Astrium.

Inmarsat 4
© Astrium / C. MÈriaux / 2004

identifying common elements across multiple projects, we were able to establish clear ownership and traceability in our code, rather than relying on people working on Project A or Project B in isolation.

Now that we have access to a central reuse library, we don't have to trawl through files scattered on disk, looking for a function or library that may or may not exist. Instead we can go straight to our LabVIEW palette, where we can get information about the VI and then use it immediately.

At one point before we started using VIPM, we found *seven* versions of the same power meter driver in our code base, representing probably six redundant weeks of effort to build and test. With VIPM, our code is all wrapped up in a package, in one place, with a recorded history and a single owner to prevent this duplication of effort. This single use case easily paid for VIPM Enterprise!

Time to market is another crucial metric for us. In order for Astrium to achieve our business targets, we have to turn around test solutions that are more robust (by reusing proven/validated code), and we have to be able to turn them around more quickly (so we need to know what state the code is in and find it quickly). VIPM helps us achieve our targets, because now developers can simply pull code off the palette knowing…

1. The code has already been verified
2. Who wrote it and when
3. That they have the latest version

This information helps them trust the code and focus on the new development at hand.

Additionally, our projects have a very long lifecycle. Since we started using VIPM, we haven't had a project "end" yet. We'll see even more benefit at that point, when we get to use VIPM's VI Package Configuration (VIPC) capability. Once a project is finished, we can archive it with its VIPC file, move on to other projects, and still recreate the exact development environment at a later date. We know this feature will be invaluable in helping us test and support our existing products in the field.

We feel like we've just started down the road with VIPM, but it has already eased the path for sharing code among developers, saving us substantial time, improving our development process, and reducing risk by allowing reuse of tested VIs. I think it's an almost perfect solution for any team that needs a central VI repository to eliminate duplication of effort, provide traceability, and clarify product ownership.

"I think it's an almost perfect solution for any team that needs a central VI repository to eliminate duplication of effort, provide traceability, and clarify product ownership."

– Chris Roebuck

VI Package™ Manager

VI Package Manager makes it easy to download and install LabVIEW Add-ons, create your own commercial add-ons, and share add-ons with your coworkers, customers, and the LabVIEW community.

Visit jki.net/vipm to learn more!

NATIONAL INSTRUMENTS™ Certified Alliance Partner

JKI is a National Instruments Certified Alliance Partner.

P.O. Box 2846
Walnut Creek CA 94595
888.891.7821
jki.net
info@jki.net

jki.net/blog
facebook.com/JKISoftware
twitter.com/JKISoftware