Sonatype

Real World Experience: Gil Clark

Gil Clark

NEXUS PRO & NEXUS LIFECYCLE USER

Using Nexus to build a more efficient software supply chain

A discussion with Gil Clark (former Software Architect, Intuit)

At a Sonatype User Group in Palo Alto, CA, Karen Gardner, CMO of Sonatype, interviewed Gil Clark (now formerly a Software Architect for the Payments Division at Intuit) about his experiences with fast-paced software development and using Sonatype's Nexus Repository Manager and Nexus Lifecycle (formerly Component Lifecycle Management - CLM). Here are excerpts from that discussion.

Gardner: What are your development priorities at Intuit?

Clark: It's really to deliver more features faster with quality. Intuit is a very competitive place. We've got to deliver a lot of features. The product managers are pounding on the teams to get stuff out. Bottom line is we've got to get out features as fast as we can but they have to work really well.

Gardner: Tell me about the development challenges you've faced.

Clark: In the past, developers had the freedom to do a lot (without standards) and did too many things in many different ways. It was a great system but it didn't favor convention. All of our artifacts were checked in and we had no way of knowing their pedigree. It was just all JARS. We had other different applications and services and almost every one was built a different way. That was untenable; too many mistakes were being made. Stuff wasn't getting into production quickly enough. We weren't able to test it automatically.

Gardner: How have Maven and Nexus helped you get organized?

Clark: I picked Maven to get everybody squared up. Maven forced the convention and, after two years, we do every build exactly the same way over 100 products. We have five DevOps engineers who can dive into any of the 100 projects, know exactly how they work, make the fix, and add the plug-in. That squared up our world. What we also did was put all of our third-party components as well as our own into the Nexus repository.

I've always liked Nexus. It does a good job. Being able to put all our versions of our components on the shelf, version them and have them reusable in that way was huge because it cleaned up some of the huge builds we had. We now know the pedigree of thousands (of open source components).

Gardner: How else has using Nexus improved productivity?

Clark: Before, we didn't have a consistent way of getting stuff to the production server. Usually it was zipped with no particular organization. Then we decided to standardize on Red Hat Package Manager (RPM) as a Linux shop. We always build on RPM, no matter what we're deploying, whether it's configuration files, properties files, or libraries.

Nexus has this awesome feature where you deploy it in the usual way like any other component but you just tag that repository as a yum-supported repository. We can manually grab RPMs just by using a normal Nexus download and we can do a Yum install. That's been really convenient. It doesn't matter what we're putting in those RPMs.

Gardner: How is Nexus helping you transition to continuous delivery?

Clark: When we decided we were going to do continuous delivery, we picked a couple products and really ran that through the automated test flow. We realized that when we got done, we were sitting on a snapshot and you can't put that into production. When you check in, you've got to have a releasable component at that moment. Then it goes through your test cycles and then it goes in the product.

In the Pro version of Nexus, there's a feature called "staging." So now anytime someone checks something in, we publish it as if it was a real full-pledged releasable component and it goes into staging. That staging form is also processed, and when we're done, it can go in production and we can promote the stage into the real deal later. Or we do it automatically on the fly. If it fails the test, we throw away the stage and we start all over again. (This process) is really allowing us to make that transition and still sort of preserve some of the old ways that we do things.

"I've always liked Nexus. It does a good job. Being able to put all our versions of our components on the shelf, version them and have them reusable in that way was huge because it cleaned up some of the huge builds we had. We now know the pedigree of thousands (of open source components)."

We let the teams be as flexible as they want during the day with snapshots. Our core services release every two weeks. Then the orchestration services on top of that might release more frequently because they're reacting to UI changes. The JavaScript UI changes might release every day. We used to promote branch. We promoted the code and then on the promoted branch you could go to production. Now, we promote the component, which is really helpful. There's only one branch that's a continuous delivery branch.

> “Nexus has this awesome feature where you deploy it in the usual way like any other component but you just tag that repository as a yum-supported repository.”

Gardner: Another priority you've mentioned is risk management when using open source components.

Clark: We're using tons of open source, probably more than 80%. And we had a (manual) process when the engineers decided to use a library. They did some research, they found the function they wanted, they found the library they wanted to use, then they'd have to go to a database and enter it and check, "yes, I've done due diligence." That's it. They didn't go check the 50,000 dependencies of that thing. They really had no way of checking all the other open source components that they unintentionally just consumed.

(On one large project) our vice president at that time said, "What? How do you guys not know what your pedigree is?" He's a very process-oriented guy. I wasn't going to track every single thing down one by one. I knew about the Sonatype CLM product (now Nexus Lifecycle) and I just said, "Well, that's my solution."

Now that (we are using Nexus Lifecycle), the light bulbs are starting to go off. Now the chief risk officer is interested and the chief security officer is interested. The awareness is growing.

Gardner: How does our product help you make better decisions about the open source you use?

Clark: We tied it into Jenkins. Any Jenkins job that has violations will show up in Jenkins, and that's sort of that first flag. That project will get a little icon that it's got a problem, and then you can trace that to Jenkins, which traces back to that project in the Sonatype product. What the product reports to Jenkins is actionable and that helps. We even promote that alert to our SDLC dashboard so the director for that project is going to see the flag that's actionable.

“It’s just that simple. It’s easy to adopt and it just works. I think we had a day of training. It really is pretty easy to use.”

The developers have the IDE integration and (if there’s a flag) the lead developer gets a task on their sprint plan. If it’s a critical alert, they don’t get to wait. In the IDE it will show if there are any alerts attached to a particular version. They just pick a version that doesn’t have an alert on it. If you modify your POM to pick a new version, the report changes because dependencies changed. Then you can look across that whole bill of materials in your project and make sure there are no threats. You can also configure your Jenkins plug-in to fail a build if there’s a threat.

Gardner: If your policy prevents you from using something it, you can’t release it?

Clark: That’s right. The build would fail.

Gardner: So if you were to summarize what you’ve done with our product, how would you say it has improved your process and the outcomes for your organization?

Clark: It just works. We got the system installed. We got it running. We modified our policies. The results show up in Jenkins and people take action and that’s it. We didn’t have to educate people. We didn’t have to document stuff. When we do demos internally, we just show the UI. We don’t have to explain a lot of context because they start to see what’s going on in the product when they look at it. It’s just that simple. It’s easy to adopt and it just works. I think we had a day of training. It really is pretty easy to use.