FEATURE

How containers change everything

The idea that everything is code isn't new, but devops is making it real

By Simon Bisson | Follow

InfoWorld | Jun 25, 2015 11:35 AM PT

Modern application development owes much to the growth of the devops movement and the automation tools it has delivered. Instead of just writing code, developers now need to think in terms the tools they're using -- and how they fit together in a flow from initial idea to a live application.

The container is one of the most important new tools in this new workflow. Technologies like Docker let us capture key services and abstract them away from underlying infrastructure. It's an approach that lets us rethink how we deploy applications and how we can take advantage of cloud infrastructures.

The whole enchilada

At a recent Amazon event in London, an AWS user described how his team handled application updates not by pushing a single change, but by making the output of a build process a "complete infrastructure."

Only when that infrastructure was deployed and tested would he switch over the DNS to make it a live system. Among other things, this approach lets you treat your old virtual infrastructure as a backup during the first few days of operation before deleting it.

The idea of pushing out complete infrastructure may seem odd at first, but when you consider the economics of cloud deployment, it's not much more expensive than deploying updates. It also means that you're deploying a known state, not updating servers and services that may have been operating for some time, and that may have had either the OS or software automatically updated.

There's no need to invest in hardware. You can use the same cloud platform for dev, test, and production -- all you need is separate virtual networks for each environment, along with appropriate access control. You're even able to work with production data in development, simply cloning stores when you need clean data.

All-encompassing containers

Containerizing applications in Docker makes it easy to abstract key application elements from your infrastructure. While it makes good devops sense to treat software this way, you're also making it easier to scale services with changing demand. Wrapping a Node.js/Seneca microservice implementation in a container makes it possible to quickly deploy new instances, either on the same host or in new virtual machines, as needed.

This approach has led to an interesting devops model: the idempotentcontainer. Instead of making an application or a service the endpoint of a build, you're building containers that wrap applications, services, and all their dependencies. Any time you make a change, you build a new container; and you test and deploy that container as a whole, not as an individual element. It's an approach that makes a lot of sense, removing some risk from the development process. In a traditional build model, it's easy to take a shortcut and just test changes, not the whole.

Once a container has been built and deployed, it shouldn't change until a new one is deployed. Because a container is a sandbox, the only way to interact with its contents is through APIs or (for an end user) through whatever UI it offers. That makes it an ideal abstraction for a microservice, where service APIs are the only touchpoints. With API definitions best thought of as a contract between devops teams, containers running on small server instances, such as CoreOS or Microsoft's new Nano Server, become a standard infrastructure block.

Going with the flow

So it's no surprise to see the Jenkins build pipeline tooling adding Docker support. Jenkins has become a standard build tool in many build processes. Its customizable modular architecture makes it easy to tune for your specific workflow, and to integrate with source control tools and with development and test platforms.

As Cloudbees CTO and Jenkins project founder Kohsuke Kawaguchi said at the event, adding Docker support to Jenkins makes a lot of sense: "It drives demand for Jenkins, treating Docker as an executable package format. You can compile and package into a binary blob that you can then take and run, and dispose when it's no longer needed."

It's clear from Kawaguchi's comments that Docker and other container formats fit neatly into Cloudbees' vision for Jenkins, "You can use it for test, for production. Fail a test, and you rebuild. You can compile code into a module, like a Ruby gem, and then down to a container, and send to Puppet for deployment."

This approach makes sense as part of an overall devops strategy, where everything, from infrastructure on down, is code. As Kawaguchi put it, when everything is code, "Git and Jenkins are the hammers for the code nail."

current lingua franca for the container world, it's good to see the Linux Foundation sponsoring container format. This initiative has brought many container developers and vendors together, oft. With a common container format with wide industry support, we'll be able to deliver containers to multiple cloud providers (both private and public) from the same build process.

A common container format won't solve all the problems associated with managing different cloud infrastructure definitions. But it will certainly go some of the way toward making it easier to move services between, say, Azure and AWS or between OpenStack and Google Cloud. Similarly, with infrastructure described by Puppet or Chef and managed in Git repositories, it's going to be possible to develop a translation layer that takes a generic VM and network description for an application and delivers an appropriate orchestration for whatever target cloud provider you have.

The idea that everything is code isn't new, but it's taken the move to devops to make it reality. With tools like Docker and Jenkins working together, we're now starting to see how it's likely to work in practice.

This story, "How containers change everything" was originally published by InfoWorld.

Simon Bisson — *Columnist*

Simon has worked in academic and telecoms research, been the CTO of a startup, run the technical side of UK Online, and done consultancy and technology strategy.