

Cloud Computing Proves a Natural for Offloading Time-Consuming Test and Development Processes

Transcript of a sponsored BriefingsDirect podcast on freeing up enterprise resources and developers' time by moving to the cloud for test and development.

Listen to the podcast. Find it on [iTunes/iPod](#) and [Podcast.com](#). Download or view the transcript. [Learn more](#). Sponsor: [Electric Cloud](#).

Dana Gardner: Hi, this is [Dana Gardner](#), principal analyst at [Interarbor Solutions](#), and you're listening to BriefingsDirect.

Today we present a sponsored podcast discussion on using [cloud computing](#) technologies and models to improve the test and development stages of applications' creation and refinement.



One area of cloud computing that has really taken off and generated a lot of interest is the development test and performance proofing of applications -- all from an elastic cloud services fabric. The build and test basis of development have traditionally proven complex, expensive, and inefficient. Periodic bursts of demand on runtime and build resources are the norm.

By using a cloud approach, the demand burst can be accommodated better through dynamic resources, pooling, and provisioning. We've seen this done internally for development projects and now we're starting to see it applied increasingly to external cloud resource providers like [Amazon Web Services](#).

Here to help explain the benefits of cloud models for development services and how to begin experimenting and leveraging external and internal clouds -- perhaps in combination -- for test resource demand and efficiency, are [Martin Van Ryswyk](#), vice president of engineering at Electric Cloud. Welcome, Martin.

Martin Van Ryswyk: Thank you, Dana.

Gardner: We're also joined by [Mike Maciag](#), CEO at Electric Cloud. Welcome, Mike.

Mike Maciag: Thank you very much. Glad to be here.

Gardner: Martin, let me take our first question to you. As I mentioned, we're seeing a lot of uptake in development aspects of the life cycle towards creation, test, and ultimately production for applications. Why is this such a big deal now? We've obviously had a lot of these technologies around for quite a while.

Van Ryswyk: You're right. The technology and the need have been around for a long time. Folks have always wanted their builds to be fast and organized and to be done with as little hardware as possible.



In places I've worked, we've always struggled to get enough resources applied to the build process. One of the big changes is that folks like Amazon have come along and really made this accessible to a much wider set of build teams. Large companies have big [data centers](#), but even those are filling up, and so Amazon has really made the difference.

The dev and test problem really lends itself to what's been provided by these new cloud players. The need for lots of different types of machines, the matrix problem of how you're going to test, the "burstiness" nature of it -- I might have only integration storms where I need machines -- all seems to lend itself to what's being provided.

Gardner: I suppose there have been shifts in the actual approach to development. There was a time when you would get a lot of code together and you would get in one big build, but we seem to be now in a more agile-development mode, in many instances, where we need to do almost constant builds.

More builds more often

Van Ryswyk: Absolutely. There are more builds happening more often. The days of the nightly build are pretty much gone for most of the people we talk to.

The other part of that agile approach is bringing testing into the automated part of doing the nightly build. So, it's build, test, and deploy a lot of times -- deploying either to production or to other test environments.

There is no brick wall any more to throw code over to test teams. All of this is happening as one seamless operation, and it's happening more often. So you're doing more and doing it more often. It just puts a tax on the resources.

Gardner: Mike, what are the reasons the traditional approach to this has fallen short? There are economic reasons, but on the technical side, have we missed anything?

Maciag: No. The traditional approaches of the concept of the overnight build or even to the point of what people refer to as [continuous integration](#) have fallen short, because they find problems too late. The best world is where engineers or developers find problems before they even check in their code and go to a preflight model, where they can run builds and tests on production class systems before checking in code in the source code control system.



Gardner: Martin, help us understand a little bit about what Electric Cloud brings to the table. With [ElectricCommander](#), for example, what is the problem set that we're solving here?

Van Ryswyk: [ElectricCommander](#) solves the problem of automating and making it efficient to do the entire build process. There's a lot of creative juice that goes into making products -- developer, design, requirements. There are tools to help you store your code safely. All of that is part of the creative process.

But, at a certain point, you just want it to happen like a factory. You want to be able to have builds run automatically. You want them to run at 3 in the morning. You want to run them in continuous integration style, based on a trigger from a [software configuration management \(SCM\)](#) system or before the SCM system even gets it, as Mike said. That's [what ElectricCommander does](#). It orchestrates that whole process, tying in all the different tools, the SCM tools, defect tracking tools, reporting tools, and artifact management -- all of that -- to make it happen automatically.

When you do that, resources are a part of that, and that's really where the cloud part comes in, having to run all those in parallel a lot of times. At the same time, different architectures have different operating systems. Then, you're bringing it all back together for a cohesive end report, which says, "Yes, the build worked."

Gardner: So, we're in the early innings, but it seems to me that the same complexity of needing to test across a variety of platforms, environments, stacks, and configurations is now being bumped up into the cloud.

I have to assume that there is going to be more than one cloud, although we sometimes mistakenly refer to it as "the cloud." Is that correct -- that we're going to have, in essence, the same heterogeneity complexity that we had to manage internally but now at a higher abstraction?

Managing heterogeneity

Van Ryswyk: Absolutely. [ElectricCommander](#), for example, was already allowing customers to manage the heterogeneity on physical machines and [virtual machines \(VMs\)](#). With some integrations we've added -- and I think people want to do this in general -- you can now extend that into the cloud. There will be times when you need a physical machine, there will be times when your virtual environment is right, and there will be times when the cloud environment is right.

Maciag: What we've seen is that the most sophisticated customers have been able to build private clouds to do exactly what Martin is talking about. The exciting part about the public cloud is that it's bringing those types of infrastructures into the affordability range and sophistication range of much smaller organizations.

Gardner: I'd like to hear from your experience how much of a leap it is for you to go from applying this to an internal cloud fabric to an external? Is it cut-and-dried or are there some other issues to be considered when you move from what some people call a private cloud to a public cloud?

Van Ryswyk: There are security concerns, bandwidth concerns, and intellectual property concerns, all the things that you can imagine, with starting to use the public cloud.

Having said that, what we're finding is that there are some customers who are large, legacy deployments with large data centers, who are used to doing it their way internally in the private cloud, who are finding projects of all sorts that they can do in the cloud -- for example, testing.

We may not want to put our source code out in the cloud -- that's a deal breaker for us -- but we can use 500 machines for few hours to do some load, performance, or [user interface \(UI\)](#) testing. That's a perfect model for us.

There are others, and this includes some of the larger organizations and a lot of the smaller organizations, who are just starting and really have no infrastructure to start with. They're going all-in from the beginning, putting everything in the cloud, the entire development environment, in some cases, even having developer machines with [Remote Desktop](#) or [SSH](#) in it.

They literally do their compiling and their daily work on machines that happen to physically be in the cloud. There are some larger companies that are taking smaller steps and finding opportunistic projects, and there are some that are going all in.

Gardner: Among this current mishmash in terms of how people are adopting, is there some pattern? Is it start-ups, small and medium-sized businesses, consultancies, or development shops that tend to be more in the cloud, or is that an oversimplification?

Willing to go all-in

Van Ryswyk: It's been surprising to me, because that's what I would have thought. The set you just described are the ones who are willing to go more all-in. The source code control is up there, the full build machine, all the testing is up there, and everything happens up there.

But, what I've been surprised at is the number of larger companies, with what you think of as more traditional models, who are willing to try something new for some different projects they have. Sometimes, it's because they're trying to support a remote location. Sometimes, it's because they've just run out of hardware or rack space, and that has driven them to try some experiments in the cloud.

Maciag: What strikes me as a comparison is the adoption of wireless technologies around the world. Those places that didn't have a wired-line infrastructure simply jumped straight to the wireless infrastructure. I think we're going to see the same thing in cloud computing. Those people who have private clouds and good IT infrastructures will continue to do that for quite some time, and those who don't will jump on the cloud first.

Gardner: Well, based on the notion that necessity is the mother of invention or perhaps experimentation, you're trying to make this a little easier for those folks who are willing to take this leap. Could you tell us a little bit about what Electric Cloud has done vis-à-vis Amazon?

Van Ryswyk: We've [developed an integration with our ElectricCommander product](#), which allows it to manipulate the Amazon environment using the Amazon [application programming interface \(API\)](#). They have a wonderful API. It allows you to have machines be created and destroyed, to manipulate that environment, and then [tie that to the ElectricCommander product](#), so that you can do that in a workflow.

I want to have 50 machines suddenly to do some builds. I want to put this workflow on those machines. It's done in a way that can be generic or specific. I want any machine that's available, whether it's physical, virtual, or in the cloud. I want to use it or I want to target a cloud machine for whatever reason.

That allows you to also do testing in the cloud. You can have Commander fire up 50 machines, as soon as the compile is done. So, you're linking the processes. Have all that testing run, get the results, and bring them back. We already have dashboards for results of all the different stages of software production.

Then, you destroy the Amazon environment, so you're only paying for what you use for the hours that you use it, and then integrate all that back to the job report, the full report of what happened during that build.

Gardner: Now, one of the complaints -- I suppose complaints you could call it -- that I've heard about Amazon is people say, well, "This is just building blocks. This is infrastructure services. There's not a lot of management controls or added value to the services, and so forth." It seems to me that you're stepping up to the plate as an ecology player. Is that fair? Are there other players that you're looking at to perhaps partner with?

Doing management for them

Van Ryswyk: That's very fair. That's exactly what we're doing. We're trying to make it so our customers who are already using our tools and new customers who are using our tools can take advantage of that cloud environment, without having to know too much about it. We're doing that management for them.

The 50 different API calls you have to make for setting up security, setting up storage, creating a machine, monitoring and all that, that's what our integration will do, so you don't have to worry about that. There are some other players out there, but none, I think, who have focused on the aspect that we're focusing on.

Gardner: Let me fully understand for our audience, for their benefit. Are you creating Electric Cloud services and your Commander services as a service, or is this something that you're going to be doing internally, but then applying to a cloud environment like Amazon?

Van Ryswyk: At this point we do not have it as a service offering. However, customers can deploy our products completely in the cloud and make a service out of it for themselves. Our customers already do this and run our products as an internal service, some of them with thousands and tens of thousands of users going to the clusters of computers in their private clouds that support our products.

Both products are really about "the more machines you throw at it, the faster and better we can do things." So, they have very large implementations. This allows them to either run all of that up in the cloud themselves or to bridge between the one they already have and the cloud.

Gardner: One of the things that have been also a stumbling block for a lot of companies, as they try to master software development, is making a leap from development to full production. What many analysts like myself are expecting is for applications to find themselves in on-premises environments, but then, perhaps for peak usage or for some economic benefit, to have them also running in a cloud or a cloud of clouds for a variety of reasons.

Is there something now about the whole lifecycle management around applications that your approach can accelerate, that helps bridge the gap between design, develop, test, and then ultimately go into a myriad of different production models?

Van Ryswyk: The answer is yes. Really, it's mostly in the pre-production world. The cloud has been very good for scaling production class applications in the public cloud. What we're enabling is providing the on-ramp for the development of those applications. For anything in the build, test, package, and all the way up through deployment, regardless of the toolset that you decide to use, we can manage the process, how it gets run in the cloud, and make it accessible to people that need to access it.

Gardner: So, there are stepping stones here, and, as I say, we're in the early innings. All right. What about getting back to the actual development phase itself? Is there something about going to the cloud build that helps in terms of agile and teams or distributed? What are some of the other productivity benefits, other than simply offloading the servers that this cloud model enables?

Maciag: First of all, it's again back to enabling people to do things that are otherwise hard or expensive to do.

Saving developers' time

For example, if I can run my software production, my build, and my test cycles twice as fast or 10 times as fast by using multiple machines, I can then save developers time. We know that developers are both expensive and increasingly scarce these days. So, there are measurable [returns on investment \(ROIs\)](#) in going to either a private or a public cloud environment. That saves anywhere from 20 to 20-plus percent plus from there.

The other piece that happens is, if people can properly run the agile development cycle that this enables, they can get their products to market much faster. We could talk for a long time about what the ROIs are on getting a product to market faster, but it ends up being in the 10-20x for any type of productivity improvement that you can think about.

Gardner: Martin, let me pose this question to you. We've heard about "test storms," and then we've heard about "cloud bursting." How do these two come together for a beneficial effect?

Van Ryswyk: "Test storms" is usually somewhat related to the life cycle. Maybe you have some milestones in your development process, where everybody has certain deliverables, and naturally people want to start testing at that time. It tends to clump up the testing work, where you have an integration point, you're integrating big chunks of work, and then you have to do a lot of testing before you go to the next phases.

Those really stress the infrastructure, because you have to build an infrastructure that's big enough for the biggest peak load. That's where this cloud bursting and using clouds can really help, because you can now provision your infrastructure for an average load.

When you have these short duration storms of activity that sometimes require hundreds and hundreds of computers to do the kind of testing you want to do, you can rent it, and just use what you need. Then, as soon as you're done with your test storm, it goes away and you're back to the baseline of what you use on average.

Gardner: Right. So this dynamic provisioning, this better utilization of resources makes a great deal of sense in the build and test phases, but, as I pointed out earlier, it's going to increasingly make sense in full production. Are the developers going to get ahead of the curve here, and, if so, what do the IT operations teams need to be more aware of?

Van Ryswyk: That's a really good question. I've talked to some IT teams, some development teams, and some of the cloud providers, and asked the same question. One fear is that the front-line development and test teams are just going to swipe their credit card and do this as an end run around IT.

I don't see that happening too much. I think that's just a fear. More likely what's going to happen is IT is going to have to, and will, slowly get their arms around this -- a lot like what happened with virtualization.

If you think about virtualization, it started with people running VMs on their desktop. "Hey, there's this [VMware](#) thing," or this other technology, "and I can do some things I could never do before and make some environments available to me just by running this VM on my machine. That's great."

More sophisticated servers

If you look what's happened over time, that's been adopted by IT and now, with server virtualization, IT organizations all over are putting in more sophisticated enterprise class servers that can handle virtualization at a much higher level.

That's the same thing that's going to happen with clouds. There will be some people who will experiment with it, and some smaller groups will get it going, but IT is going to look at this and say, "Wow, this really solves the problem we have. We can't meet that peak demand. We would like to have some better service for a quick turnaround."

If someone wants a machine, it's not two days before you get the machine. It's two minutes before you get the machine. Those are the kind of things that their customers will ask for, and they will figure out how to work with.

Gardner: So, it's no surprise that Amazon is happy to partner with folks like you, because I think the development and test use case could become in fact the adoption path for many organizations into larger cloud adoption. If the IT department recognizes that they're going to satisfy the needs of their developers by adopting cloud -- both internally and externally -- they get some competencies added. Then, they start applying that into full production. Does that make sense?

Van Ryswyk: Absolutely. From analyst reports we've read and from talking to Amazon themselves, what we've heard is that dev and test is the biggest user of the cloud right now. There are some [Web 2.0](#) type applications that are very public and they're running on production application on the cloud, but the majority of the use cases are folks in dev and test using the cloud.

Again, that's a lot like [virtualization](#), which really started with a lot of QA teams using VMware to begin with. It's a natural fit. It's what people are already doing with the cloud, and we're going to provide tools here to make it easier to do that.

Gardner: How do you get started? If you're a developer or even an operations IT individual, you're listening to this, and it starts to make some sense, is there a high wall that you have to get over in order to start, or is there a path that sort of gets more towards that crawl-walk-run approach?

Maciag: The simplest way is, when you begin, you begin with an infrastructure that replicates what you do today. Certainly, doing what somebody does today, whether it be just a simple build-and-test cycle or a simple continuous integration cycle, is not difficult to replicate in a more robust environment.

Replicate what you're doing

If you replicate that in a more robust environment, like ElectricCommander with access to the cloud, it's now easy to turn things on like, "Let me take that test that I was running serially and make it run in parallel. Let me go grab 10 or 100 or 1,000 cloud resources to go run. Let me create a pre-flight build environment rather than simply continuous integration environment."

So, I always recommend that customers start with replicating exactly what they're doing today and then they can evolve into all kinds of sophisticated things right after that.

Gardner: Martin, anything to offer on getting started, either from an education or a practical approach?

Van Ryswyk: Yes, for nuts and bolts, I'd say, use two websites, ours and Amazon's. You can learn a lot about what we talk about, about the software production, management life cycle, and what Commander can do for you from our website. Then, check out the [Amazon AWS website](#). I don't mean to just pick on them, there are other cloud providers, but Amazon is out in front. They're just a little bit ahead of some of the others. They have great resources.

So, you can understand the boundaries of how security works. They've got a great security white paper. How does this all really work? How do I get charged for this? Get your brain around the big parts of it, and then find a tool that will help you, so you don't have to get into the fine parts of it.

Gardner: You mentioned the charging aspect. That is important. Is there a shift here also, as we move from traditional test and dev to a cloud or a hybrid approach that changes how the charging and paybacks work?

Van Ryswyk: It does. I guess I could give you an example. Here in Electric Cloud, because what we do is build the kind of products we do, we're very sensitive to the build time of our own builds. We had a test that we were running that was really taking too long and the developers didn't want to wait for that for every build, but yet we wanted it to run every build to ensure the quality.

As an experiment with cloud computing, we decided to see if we could put in our process, offloading that one test. It happened to be for our reporting module, so it required installing a database and a web server and exercising the whole product as a system test with some fake data to make sure that the reports came out accurate, as we expect them to. It takes about three, four, five minutes to run -- even that we're sensitive to here.

When we looked at the economics of it, we said, "If we put this out at Amazon, it ends up being about 35 cents a test." Is that worth it to us?

We were considering dropping that test from our portfolio of test because of the time it was taking, but if we can offload it and do it in parallel up at Amazon while the rest of our stuff was running here, would that be worth it? It was a no-brainer. So, 35 cents a test to get that kind of value and validation out of our product? Absolutely. Those are the kinds of economic decisions we make.

Gardner: Well, great. I'm afraid we're about out of time. We've been discussing how to use cloud compute technologies and models to improve the test and development stages of application's creation and refinement.

We've been joined by two gentlemen from Electric Cloud. Martin Van Ryswyk, vice president of engineering, and also Mike Maciag, CEO. I want to thank you both.

Maciag: Thank you, very much.

Van Ryswyk: Thanks, Dana.

Gardner: This is Dana Gardner, principal analyst at Interarbor Solutions. You've been listening to a sponsored BriefingsDirect podcast. Thanks for listening, and come back next time.

Listen to the podcast. Find it on [iTunes/iPod](#) and [Podcast.com](#). Download or view the transcript. [Learn more](#). Sponsor: [Electric Cloud](#).

Transcript of a BriefingsDirect podcast on freeing up enterprise resources and developers' time by moving to the cloud. Copyright Interarbor Solutions, LLC, 2005-2009. All rights reserved.