PhUSE 2012

Paper AD02

Testing in the Cloud. Not just for the Birds, but Monkeys Too.

Geoff Low, Medidata Solutions Worldwide, London, United Kingdom

ABSTRACT

Cloud-computing is here and has been for some time. Outdated is the concept of your walking into a data centre, pointing at a server and saying, "there's my data." There may be a feeling of submission of control, but this is more than compensated by the power it brings. The cloud makes your data available to you anywhere, using all the same security controls that you'd expect from your "big iron" data centre. Instances become what they should have always been, a utility that is there when you need it, which can be powered up and down according to you or your customer's demands. And when you don't need it – it's gone. This talk will cover the utility of the cloud as a platform for testing applications and how Medidata is using it to develop a system for improved resilience across a platform.

INTRODUCTION

Resilience is a property of a software application that describes how it performs over time and operating environments. Avoiding performance issues in production environments requires that a software development company factor resilience testing into its platform development process. Resilience testing will be complex in a multi-product platform environment, with multiple inter-dependencies necessitating a complete copy of a real environment to accurately mock production. This will become expensive to host and difficult to maintain, so we look for alternatives that give us the ability and power to run resilience testing.

This paper will discuss how the use of cloud computing environments add to the process of agile development, testing of software and how it opens up the testing environment to platform level testing.

The Cloud

The cloud or cloud computing is the logical extension of viewing computing power as a utility—much like power and water. Although it is considered that we are now in the era of cloud computing, the origin of the cloud itself dates back to the mainframes of the 1950s. Organizations used time-slicing of their computational resources to aid in a return on investment for the large cost involved in purchasing the machines, by on-selling access to the resource. [1]

The cloud computing environment is now a multi-billion dollar services industry.[2] It was revolutionized by Amazon, which identified that within its extensive data centres they were only utilizing 10% of its capacity. They looked to commoditize this surplus resource and this lead to the creation of Amazon Web Services (AWS).[3] Other notable solutions in the marketplace are Microsoft Azure[4] and Google App Engine.[5] In parallel with and in cooperation with the prevalence of cloud-computing are infrastructures developed to be built on the cloud. Some examples are Infrastructure as a Server (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).[1] Software as a Service has gained significant mindshare across the information technology industry and it is primary operation mode of Medidata's products.

The initial definition of cloud computing constituted internet-provided resources; over time an intermediary platform has evolved, which takes the infrastructure from the cloud and implements it within a company network. The internal or private cloud computing environment takes the underlying concepts (many servers, extensive virtualization) and deploys it within a company's network/data centre, such as that provided by VMWARE's vCloud product[6]. The use of private clouds gives an organization control over its

hardware – but comes at the cost of the volume discounts that cloud providers can leverage. Cloud providers have implemented Virtual Private Clouds (VPCs) to provide security for organizations that require private network infrastructure in the cloud. [7] In addition, users of the cloud can mix and match provisioning – for example using a dedicated (or reserved using cloud parlance) Database server instance and cloud instances for Application or Web servers.

In addition the cloud includes the use of availability zones to make the data available in many places worldwide. The use of availability zones adds to the redundancy of the system with multi-data centre fail-over scenarios becoming possible at minimal extra cost. Availability zones create opportunities for regionalization of internet resources, reducing latency. Backups are also be made to different locations in the world and to different media for cost-effectiveness. There is a lot of power that the cloud carries with it.

The Clinical Cloud

Despite the considerable cost-savings and flexibility of the cloud, the pharmaceutical industry has been conservative on its adoption. There is a degree of insecurity within the industry that keeps auditors visiting data centres and checking the same sets of locks, access logs and security credentials despite knowing that most of these institutions are certified with industry standards such as SAS70[8]/SSAE 16/SOC 1. The "point at my server" mindset is a traditional one, but is one that we should look to re-evaluate.

We are operating within a highly regulated environment and must consider at all times the rules regarding the capture and keeping of clinical data as mandated by the regulatory authorities. The FDA Predicate Rules are defined as part of 21 CFR Part 11. The predicate rules require that the following information is known at all times:

- Hardware Serial Number
- System Configuration
- Equipment Location
- Exact versions of all Software.

When looking to satisfy these predicates, the requirements for "Software Configuration" and "Exact Versions of all Software" can be addressed by strongly controlled build environments (such as those provided by remote configuration providers including as Opscode's Chef[9] and Puppet Labs's Puppet[10]). Remote configuration providers are based on top of source code management (SCM) systems (such as git[11]), that satisfy the Who-What-When questions for auditors.

The remaining two criteria can be more challenging to resolve - availability zones may include a number of "physical" data centres. The instance can be a virtualized so may be on one of many servers and as such the definition of a Hardware Serial ID becomes more challenging.

One possibility would lie with the definition of a Hardware Serial ID being amended to refer to a universally unique identifier (UUID) that may not be tied to an underlying hardware component. Cloud Providers already use VPC resources or dedicated machines to ameliorate concerns around these predicates.

Amazon has recently produced a guidance document illustrating the facilities of AWS that would comply with HIPAA/HITECH rules pertaining to protected health information (PHI).[12] In addition, Amazon has recently announced Amazon Glacier, a low cost storage vault that included the storage of "experimental drug results" as a use case – such as genomic data.[13] The data is encrypted and stored on special low power servers, and can be requested to be retrieved on demand.

An example of actual cloud environments for sensitive data is the AWS GovCloud for the hosting of Federal Data – it is built to meet all the defined security requirements (including FISMA, SSAE 16/SOC1 (formerly SAS-70 Type 2), ISO/IEC 27001, and PCI DSS Level 1)[14]. The FDA is embracing cloud-computing with the internal development of a private cloud and continual evaluation of the public cloud including security and economic assessments.[15] The current CIO has stated that it is an imperative of his to adopt the cloud to make more clinical data available for people to be able to mine, and the only way that can feasibly be attained is looking at a cloud-based solution.[16]

Testing in the Cloud

Continuous integration (CI) testing forms a fundamental part of our development methodology at Medidata. As an Agile development company, the ability to get continuous feedback on the effects of code changes on the entire codebase is invaluable. One such tool for CI is Jenkins.[17] Jenkins is an expandable CI tool that uses plugins to customize and extend the platform according to requirements, covering areas such as federated login and SCM integration. We use GitHub for hosting our source code.[18]

We have a nightly continuous integration task that runs on the development branch. We use the GitHub integration plugin to start a new CI run when source code commits are Pushed to GitHub. The workflow is shown in Figure 1.

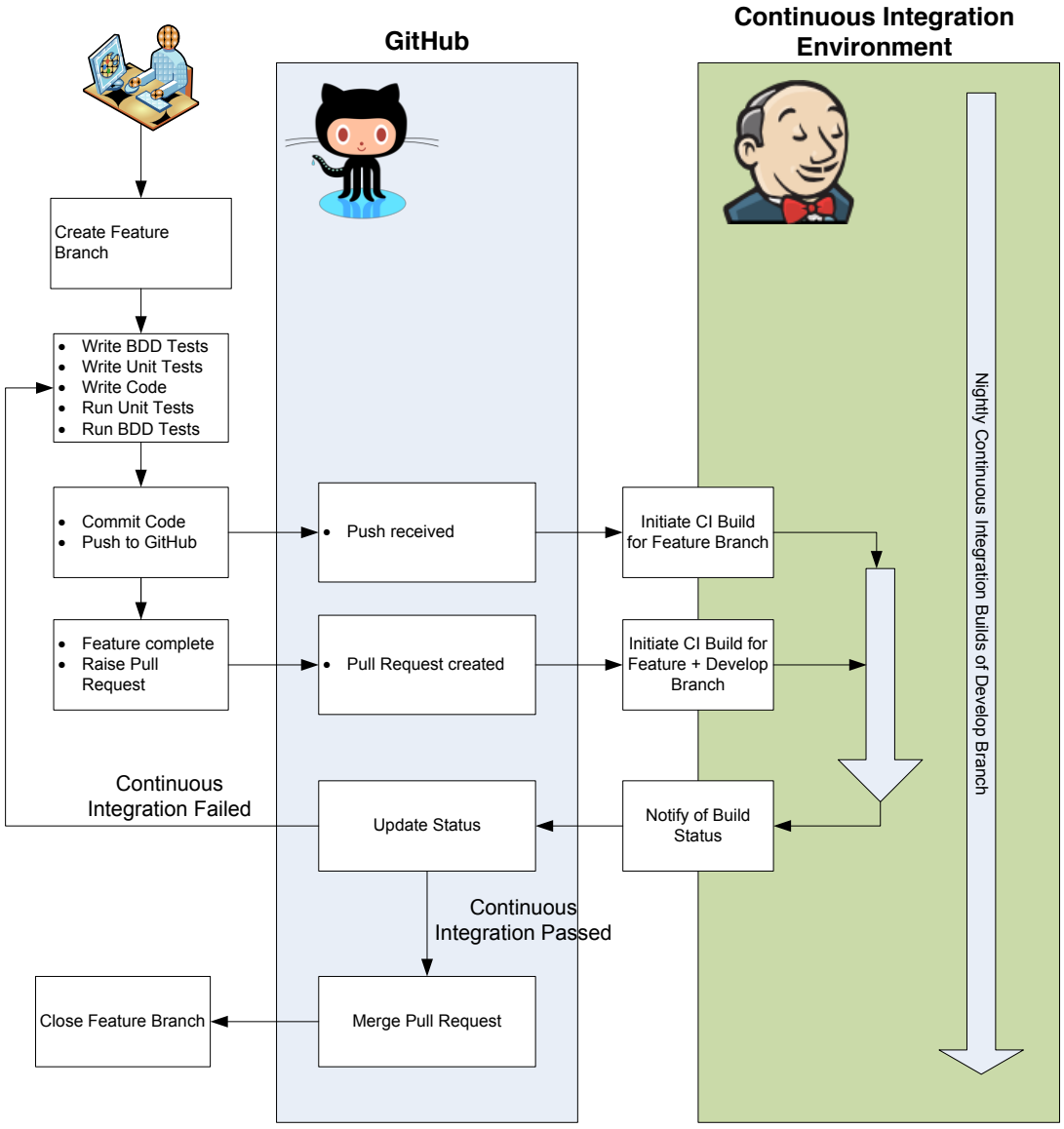

Figure 1: Integration of automated CI tasks into an Agile Development Process

One issue we're finding with the Automated CI on Push process is being a bit more selective on when a run takes place. In the middle of a sprint, often features that aren't complete get pushed (say for the scenario where we want code review to take place) - we know that changes made may break existing code tests. Ideally, we'd like these commits to get sandboxed. We are evaluating options to resolve this, including defining flags for work in progress or providing metadata to the CI server.

This usage of a Continuous Integration environment fits very well into our Agile Development Process. Different features are developed in parallel, with a continuous integration process on the development branch. Developers are notified immediately when they have implemented a change that affects other areas in the code adversely, leading to fixes being implemented as soon as they occur.

GitHub has recently announced the Status API[19] that allows tools (such as the CI server) to post status updates back into GitHub. An example use case is the annotation of a Pull Request (a Pull Request is the method used by GitHub to wrap the process of merging of code between branches and forks) with the outcome of the Continuous Integration build. The peer review of Pull Requests into the development branch from the feature branches can then be viewed for issues across all code, rather than just the set of changed files in the Pull Request. There will be an annotation in the audit trail stating that a CI job was run, and the outcome of the CI run.

Nothing will slow an agile sprint more than developer downtime. Waiting for the tests to run is an example of downtime that can't be avoided. Test servers (or continuous integration slaves) are provisioned within the network as a finite resource and, given a typical sprint, there are a number of developers working on different features in parallel. Each of these features will all need access to the testing resource and often a queue will arise for the testing servers. In addition, continued use of a shared test resource can accumulate code detritus after repeated use. This will result in the performance of the test servers decreasing over time and as a result, longer time taken per test run. At some point, it becomes necessary to refresh the test server with a complete reinstall – leading to downtime.

A virtualized test environment is a solution, such as VMWARE's Lab Manager.[20] A set of template virtual machines are defined, that can be rolled out as blank test environments for Development teams to use, and dispose of. This will help resolve the clean room implementation and will provision more instances per set of testing hardware.

The cloud can be used as an alternative to the costs of buying and maintaining a dedicated virtualisation server - using the same build and deployment processes. Unlike a dedicated internal virtualisation server with no demand, there are no instances. The Continuous Integration server can be configured to slave according to resource rules - such as using internal servers until capacity is exhausted and then start creating slaves in the cloud. Given the choice of an internal versus external build slave, the innate latency will preference internalized systems over external ones.

The general utility of the cloud environment, along with configuration- and build-automation tools makes the cloud an excellent platform for distributed testing of software. The developer no longer joins a queue for the testing system, instead they "spin up" a new instance to run tests, capture the logs and then "tear it down." In each case the platform is clean and is configured to an exact set of requirements and is only available for as long as required.

The Chaos Monkey

Continuous Integration is a vital component of our application development, and in some way it contributes to resilience testing in the whole. However, CI testing is limited to a single product, rather than an entire platform. We need a way to incorporate testing across multiple products, or a platform. To start off we'll consider a single web application.

We consider the case of a three-tier application show in Figure 2.

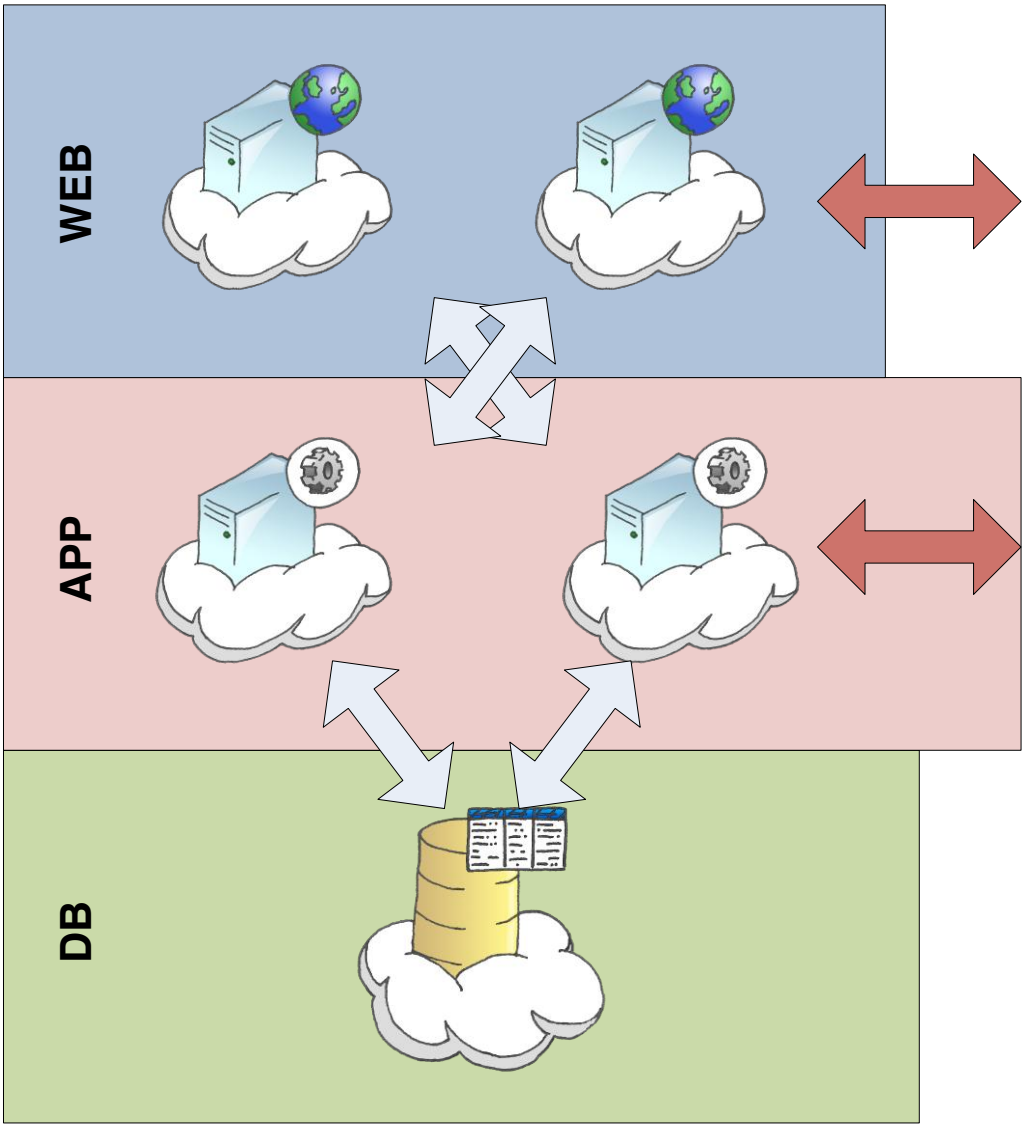

Figure 2: A three tier Web Application

Given the Web application illustrated in Figure 2, we consider what happens as the load scales up. Each instance in the web- and application-tier will be handling more requests until the resources are overloaded and the web application as a whole becomes less responsive.

One solution for resource management is to use horizontally scalable implementations – rather than creating singular large application- or web-tier instances, the architect defines a pool of small instances, with the ability to automatically increase or decrease the number of instances in the pool dependent on demand.

This works well for e-commerce companies, as they can manage their infrastructure to be responsive to demand, such as online retailers around Christmas. One well-known company already doing this is the online video retailer, Netflix—a heavy adopter of AWS—has built a number of services around the auto-scaling function of AWS.[21]

Netflix, in order to maintain overall system resilience, built a system around planned and unplanned failure–as stated "the cloud is all about redundancy and fault-tolerance".[22] The system that Netflix built is called the Simian Army, with the first iteration being called the "Chaos Monkey." The Chaos Monkey randomly terminates instances and services within an auto-scaling group—as they described it, "unleashing a wild monkey with a weapon in your data centre (or cloud region)". Engineers can then look at the effect of issues on the platform as a whole, rather than focusing deterministically on specific

scenarios for failure. By factoring failure into the system, it allowed them to have information about the effects of a single system failure on the platform as a whole and manage its infrastructure and processes around it. Netflix has subsequently advanced the Simian Army project to use monkeys that introduce artificial latency into the platform and find and destroy non-conformant instances.

It seems a strange paradigm, deliberately inducing failure in a production system. We don't propose that we are doing this procedurally with live data, like an online video retailer—its data are much less precious than ours. However, having a system deployed in a validation environment and trying deliberately to break it is valuable exercise in what we might expect in the production environment (over and above the considerable testing that we already undertake).

We are actively creating our own Simian Army to march upon the validation environment for Medidata's Clinical Cloud—something that has been helped by the open-sourcing of the project by Netflix[23].

CONCLUSION

The cloud is here, and it is an invaluable platform for the software industry. Despite the relatively low adoption rates of the public cloud in the pharmaceutical industry, we are confident that services give the levels of redundancy and flexibility to bring power and cost savings to those adopting it. We believe in the power of the cloud, and we are continuing to develop and use it as part of our clinical cloud platform.

REFERENCES

[1] Cloud Computing (Wikipedia – retrieved Sep 2012) - http://en.wikipedia.org/wiki/Cloud_computing
[2] Roundup of Cloud Computing Forecasts and Market Estimates, 2012 - http://softwarestrategiesblog.com/2012/01/17/roundup-of-cloud-computing-forecasts-and-market-estimates-2012/
[3] Amazon Web Services – https://aws.amazon.com
[4] Windows Azure - http://www.windowsazure.com/en-us/
[5] Google App Engine - https://developers.google.com/appengine/
[6] vCloud - http://www.vmware.com/products/datacenter-virtualization/vcloud-suite/overview.html
[7] Virtual Private Cloud (Wikipedia – retrieved Sep 2012) - http://en.wikipedia.org/wiki/Virtual_Private_Cloud
[8] Statement on Auditing Standards No. 70: Service Organizations (Wikipedia – retrieved Sep 2012) - http://en.wikipedia.org/wiki/Statement_on_Auditing_Standards_No._70:_Service_Organizations
[9] Opscode Chef – http://www.opscode.com
[10] PuppetLabs Puppet – http://puppetlabs.com
[11] Git SCM - http://git-scm.com
[12] Creating Healthcare Data Applications to Promote HIPAA and HITECH Compliance (Aug 2012) - http://d36cz9buwru1tt.cloudfront.net/AWS_HIPAA_Whitepaper_Final.pdf
[13] Amazon Glacier - http://aws.amazon.com/glacier/
[14] AWS GovCloud (US) - http://aws.amazon.com/govcloud-us/
[15] FDA IT and Informatics Transformation - FDA Science Board – May 2, 2012 - http://www.fda.gov/downloads/AdvisoryCommittees/CommitteesMeetingMaterials/ScienceBoardtotheFoodandDrugAdministration/UCM302749.pdf
[16] FDA's CIO pushes for open source, cloud computing - http://www.fiercebiotechit.com/story/fdas-cio-pushes-open-source-cloud-computing/2012-04-25
[17] Jenkins Continuous Integration Server – http://jenkins-ci.org
[18] GitHub – http://www.github.com
[19] GitHub Status API - http://developer.github.com/v3/repos/statuses/
[20] VMware Lab Manager - http://www.vmware.com/products/labmanager/overview.html
[21] Auto-scaling in the Amazon Cloud - http://techblog.netflix.com/2012/01/auto-scaling-in-amazon-cloud.html
[22] The Netflix Simian Army - http://techblog.netflix.com/2011/07/netflix-simian-army.html
[23] Chaos Monkey Released Into The Wild - http://techblog.netflix.com/2012/07/chaos-monkey-released-into-wild.html

ACKNOWLEDGMENTS

I would like to thank Management at Medidata Solutions Worldwide for giving me the time to research and assemble this paper.

RECOMMENDED READING

- Cloud Computing—Driving Security, Performance and Quality (http://mdsol.com/documents/library/cloud_computing.htm)

- Four Reasons We Choose Amazon's Cloud as Our Computing Platform (http://techblog.netflix.com/2010/12/four-reasons-we-choose-amazons-cloud-as.html)

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Geoff Low
Medidata Solutions Worldwide
Harford House, 1 George Street
Uxbridge / UB8 1QQ
Work Phone: +44 (0) 1895 275704
Fax: +44 (0)1895 275602
Email: glow@mdsol.com
Web: http://www.mdsol.com