WHITE PAPER

Automating Continuous Integration over Complex IT Infrastructure

Quali

"The purpose of Continuous Integration (CI) is to avoid a long, painful, waterfall integration process that wastes time and causes uncertainty in software development projects"

Introduction

Continuous integration (CI) is a necessary growth stage in the journey to DevOps. However, in many organizations, the complexity of the IT infrastructure that applications must be qualified against presents a serious obstacle to achieving CI automation. This paper reviews continuous integration concepts; looks at the challenges of automating complex IT infrastructure; proposes best practices for building a sustainable automation of complex IT infrastructure; and describes how to use Quali's solution for building an elastic, self-service infrastructure that can serve users and automated processes such as continuous integration.

Continuous Integration and Automated CI - A Review

The term 'Continuous Integration' originated with the Extreme Programming development process, as one of its original twelve practices. The purpose of Continuous Integration (CI) is to avoid a long, painful, waterfall integration process that wastes time and causes uncertainty in software development projects. CI dramatically shortens the integration loop to, ideally a daily process. Anytime a developer checks in code, CI practice mandates that s(he) create a build and run regression tests immediately to ensure that everything entering the mainline meets a quality baseline and doesn't compromise other developers' code. When automated, the CI process becomes a code-qualifying machine that is not dependent on developers following all the process steps manually, but which enforces daily or otherwise regular regressions on all code check-ins.

The most common way to automate the CI process is by using a CI server, such as Jenkins, TeamCity, IBM UrbanCode or others.

The IT Infrastructure Challenge to CI Automation

In theory CI automation is relatively straightforward: Purchase CI server, then start automating the CI process. CI servers assume the purchase of the underlying infrastructure, which is typically just a pool of hosts on which need to load builds and run regression tests. In these cases, a pool of virtualized machines running on hypervisors may be sufficient, or alternatively a cloud-based PaaS may serve to host the whole continuous integration process.

Complex IT Infrastructure: Deficient CI Automation

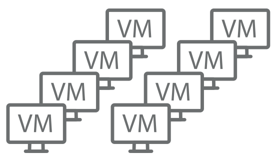
Hybrid

Networked

Legacy

Common Misperceptions

VMs don't have infinite availability



Automation ~~must~~ does'nt have to be script based

Orchestration ~~is not~~ can be intuitive

Being agile is not just automating regression processes

However, when IT infrastructure is complex, this model doesn't work, because the time it takes to assemble complex infrastructure environments prevents CI testing from happening in a rapid and regular fashion. Some examples of complex IT infrastructure include:

- **Hybrid infrastructure**
  where applications deploy across a combination of traditional, virtual and public cloud computing
- **Networked**
  where environments aren't just a pool of computing, or even a data center "stack" but are arbitrarily connected as a network "topology"
- **Legacy systems**
  sometimes application and service development and testing involves resources that aren't x86 based, such as mainframe sessions

In these cases, neither PaaS solutions nor solutions built solely on server virtualization are adequate to automate the infrastructure. As a result, continuous integration processes cannot effectively be automated.

Challenges in Automating Complex IT Infrastructure

If IT infrastructure complexity is an obstacle to automated CI processes, it's helpful to step back a pace and understand what is really needed to automate such infrastructure. One common misperception is that allocating infrastructure is simply a matter of creating provisioning scripts. The problem with this notion is that it ignores a number of important considerations:

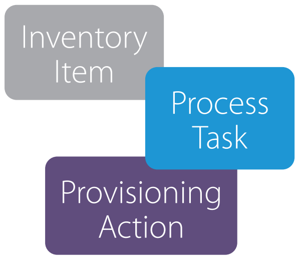
- Complex infrastructure does not have infinite availability. Even for heavily virtualized resources, there are limits to capital budgets, space, power and cooling. If resources can't be deployed in an elastic fashion, low utilization will not only cost an arm and a leg, but will ultimately block users from needed resources, reduce productivity and harm agility.

- Typical orchestration and scripted approaches to automation don't provide a very intuitive way to deal with topologies of resources. Most orchestration and automation tools grew up around compute virtualization and simply don't have constructs for dealing with the notion of network topologies. The few exceptions are almost solely based on network virtualization, but require a whole new layer of network tunnel overlay functionality be deployed before they can be of any relevance.

- Complex infrastructure may need to be shared between automated processes and human operators. Self-service is a concept that applies both to continuous integration and to users. While continuous integration

Automation Best Practices

Object Oriented Automation



Out Of the Box & Do It Yourself Tools

may have a dedicated pool of resources, it is likely that there will be both types of "users" accessing expensive IT infrastructure. So, automation needs to handle both these types of use cases.

- Complex infrastructure itself is continuously changing in terms of new software and hardware. Agility doesn't just apply to the regression process, but to the ability to maintain automation relevance against a complex stew of OS version upgrades, new blades, chassis, etc. that comprises the underlying infrastructure. Without a way to ensure timely adaptation of automation to these changes, infrastructure will block continuous integration processes. Lack of sustainability of the automation methodology itself can doom infrastructure agility over time.

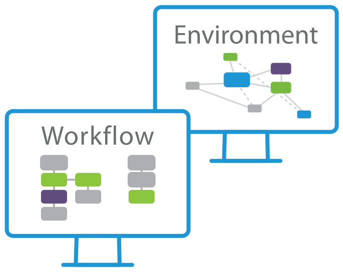
Agile Infrastructure Automation Best Practices

To counteract the challenges spelled out above, Quali recommends four best practices for ensuring agile infrastructure automation to enable continuous integration processes:

1 **Build and Maintain an Object Layer:** One of the most important methods for ensuring that automation is sustainable is to implement all automation elements as building block objects. For infrastructure automation, there are three main categories of objects:

  - **Inventory:** When infrastructure is heterogeneous and therefore a combination of "metal" components such as dedicated servers, non-virtualized storage, networking and legacy systems alongside private and public cloud virtual machines, it is critically important to create a hierarchical, granular inventory of all infrastructure. The term 'inventory' is intentionally used, rather than just 'resources' because to allow effective reservation and visibility into the state of resources, orchestration must act against the actual inventory state.
    A hierarchical library approach is important to allow nesting of sub-objects so that the granular resources that need to be deployed in an environment are distinct, reservable and configurable. Inventory objects attributes should be configured with values that allow for abstractions and to populate input parameters of provisioning and process objects.
  - **Provisioning:** These objects are used to drive discrete provisioning actions for particular inventory resources, or to construct provisioning workflows for entire environments.
  - **Process tasks:** These objects are used to construct process workflows such as provisioning validation, continuous integration and continuous delivery.

User Friendly Visual Platform



Self-Service Infrastructure

2 **Leverage both OOTB and DIY Integration Tools:** One of the major reasons why infrastructure automation projects fail is due to the inability to create or maintain sufficient interfaces to integrate with enough infrastructure to make automation relevant.
Since infrastructure changes over time, sometimes rapidly given the sheer number of different components in a complex environment, it is important to not only have access to vendor-provided, out of the box (OOTB) libraries, but to be able to use DIY tools to develop and maintain interfaces independently so that automation is not captive to and dependent on vendor roadmap timelines. DIY tools should support a number of different ways to create infrastructure interfaces:

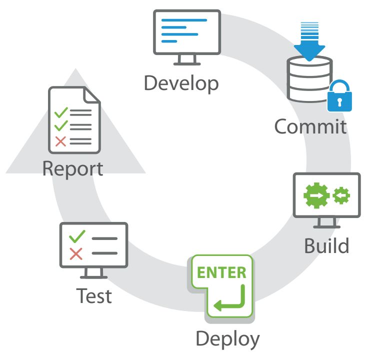
- **Record and parameterization:** This is distinct from simple "record and replay" tools which only create a simple script that must be reused in the format it was recorded in. A record and parameterization process allows the capture of CLI session and SNMP interactions, then the extraction of key input parameters as variables, to allow the recording to be made into a flexible object.
- **REST and other API's using any scripting language**, including Python, PERL, TCL, Puppet, Ruby, etc.
- **Existing scripts:** Rather than throwing away existing automation and starting from scratch, DIY tools should allow integration of existing scripts without requiring them to be changed, by using an object-wrapper methodology.

3 **Visual Environment & Workflow Creation:** One of the industry's mismatches between vendor offerings and IT infrastructure teams is that most automation tools are highly code-centric, while most IT infrastructure teams have relatively few programmers. Given the latter reality, it's important to prevent programmer bottlenecks by using GUI-based, visual, drag and drop-style tools to author both infrastructure environments and automation workflows (for provisioning and reclamation). This means that the many knowledgeable IT workers in infrastructure teams who understand the infrastructure environments and programming logic, can effectively take part in the creation of reservable environments and associated provisioning and reclamation workflows. By eliminating the programmer bottleneck, infrastructure automation can achieve dramatically better scaling and coverage of needed business processes and bring the desired return on investment (ROI).

4 **Self-Service Infrastructure for both Users and Automation Processes:** Based on the foundation of a strong object layer, OOTB and DIY integration, and visual environment and workflow creation, engineering teams can build a self-service capability around any type of IT infrastructure. This self-service infrastructure enables both users interacting through a self-service catalog, as well as continuous integration processes accessing the infrastructure programmatically. The result is a single pool of infrastructure resources that offers the greatest resource liquidity, utilization, cost effectiveness and productivity support for end-users and automation processes.

Quali

"Quali's Automation Workflow enables infrastructure agility and automated continuous integration processes"



The Quali Solution for Automated CI

Quali offers integrated self-service orchestration and workflow automation solutions that make complex IT infrastructure elastic and accessible both for human operators and automated CI processes. Quali's automation framework includes a number of important building blocks that enable infrastructure agility and automated continuous integration processes:

- **User-defined inventory inclusive of any type of resource**

  Datacenter and cloud architects can design an inventory data structure that hierarchically orders resources with the level of granularity needed to maximize multi-tenant infrastructure sharing. The inventory not only applies to cloud-based and virtualized resources, but also to traditional devices such as networking switches, non-virtualized storage and dedicated servers. For example, a switch can be structured hierarchically as chassis, blades, and ports, so that resources can be allocated down to the port level. Resources can also be development tools (SVN, CI, Build servers, etc.), application components, and logical elements such as IP subnets.

- **Powerful automation object authoring**

  No matter what the interface required to provision or control a resource, engineers can use Quali's automation tools to create provisioning and process authoring. Quali's architecture can create objects based on recording CLI or SNMP interactions, or based on scripting via Perl, Python, Ruby, TCL, Puppet, JAVA, Jscript, etc. Object wrappers can also be created for existing scripts.

- **Visual, inventory-aware environment/topology design**

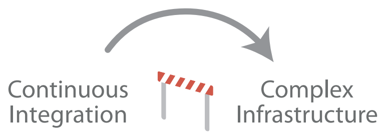
  Unlike the typical environment design cycle that begins with an abstract, Visio drawing, architects can drag and drop any type of resource from the actual inventory onto a live canvas, connect them and create a reservable, actionable environment. These environments can be arbitrarily complex, including the ability to handle any topology that is physically and logically possible in the actual data center.

- **Resource and topology reservation and scheduling**

  Every inventory resource and every environment can be reserved for a user-specified period of time subject to administrative constraints, either immediately or at a scheduled point in the future. This provides accountability, allows visibility to utilization and increases effective infrastructure utilization.

Quali

"Using the Quali platform, continuous integration workflows can "self-serve" from available environments in a programmatic fashion, just like users can self-serve from a web catalog"



"Quali removes the barrier between automated continuous integration and complex infrastructure"

- **Auto-provisioning and reclamation**

  Every environment can be paired with both auto-provisioning and auto-reclamation workflows to ensure that both set-up and tear-down happen quickly and accurately. Auto-reclamation ensures that when the reservations are ended or environments are released, that all resources are restored to a baseline configuration so that the next set-up sequence can happen successfully.

- **Object-oriented automation architecture and visual, drag and drop environment and workflow authoring**

  Quali provides a patented object-oriented approach to inventory, provisioning and process tasks combined with Visio-like environment design and drag and drop workflow authoring. Quali's visual tools maximize the whole infrastructure team's productivity, ensure the easy and sustainable maintenance of the object layer, and allow maximum automation coverage. Automation workflows can implement lower-level validation processes or higher-level processes such as continuous integration.

- **Out of the box interfaces libraries & DIY interface object authoring**

  Quali offers both OOTB libraries for many popular infrastructure products including compute, storage, networking, virtualization and public clouds. In addition, Quali offers a GUI-based DIY interface authoring tool to enable independence and maximum agility.

- **Easy to use self-service portal**

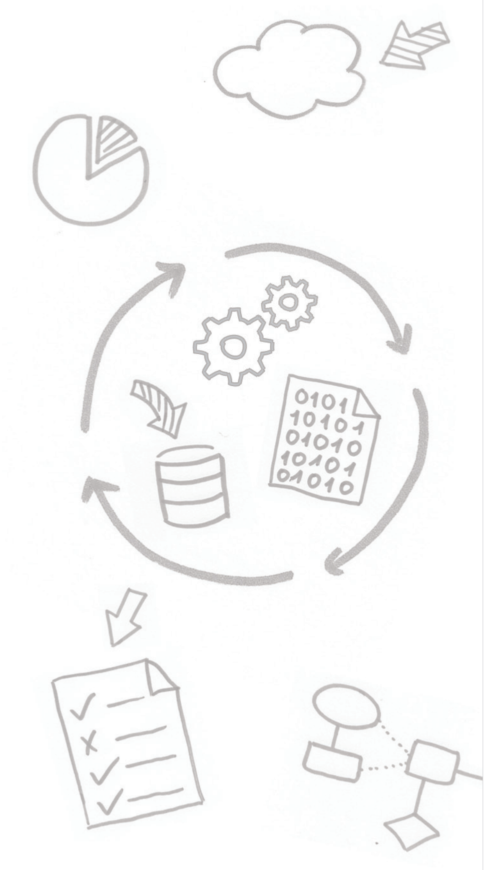
  Quali provides a powerful self-service catalog, where architects can publish completely automated infrastructure environments.

- **Tight link between continuous integration and infrastructure environment automation**

  Using the Quali platform, continuous integration workflows can "self-service" from available environments in a programmatic fashion, just like users can self-serve from a web catalog. CI processes can be completely orchestrated by the Quali platform, including monitoring for code commits and launching builds and deploys against dynamically allocated environments. CI processes can also integrate with CI servers.

- **Business Intelligence**

  Quali offers a powerful business intelligence tool that can be used to track critical KPI's for continuous integration such as defect rate trending and failure stage reporting, as well as infrastructure resource utilization rates.



Conclusion

Quali removes the barrier between automated continuous integration and complex infrastructure. By leveraging the Quali orchestration and automation platform, IT and engineering teams can achieve high automation coverage, sustainable and lower-cost maintenance of automation, and broader infrastructure team productivity, higher infrastructure utilization, and improved management visibility. The net result is significant CAPEX and OPEX savings, increased business velocity and agility, and higher levels of market competitiveness.

To learn more about the global leader in agile infrastructure automation, please visit www.quali.com

WP-CS-2

Shiri Piorovich Studio

Quali