# Cloud-Enabled Software Development and Testing: Putting the Agile into the Infrastructure

WHITE PAPER

BROUGHT TO YOU BY SKYTAP

# Contents

# Executive Summary

Agile software development is winning the hearts and minds of developers and testers in leading enterprise organizations. In the "State of Agile Development" 2011 study, VersionOne highlights that 80% of the respondents from 6,042 companies surveyed have adopted Agile development practices within their organization. Nearly 50% of respondents had between two and five Agile projects underway, and one third said their organization is running 11 or more. There is a business reason for this momentum. The Agile development model enables software teams to produce higher quality software that is more in-sync with customer needs and delivers release cycles faster and more cost-effectively than ever before.

Unfortunately, most software teams that adopt Agile development struggle to achieve its full potential due to legacy IT challenges, especially in the enterprise. While the Agile model accelerates the software development process, many teams find that their IT environments are not optimized to support the full potential of their Agile development release cycles. These legacy environments are often too slow, inflexible, and inadequate for Agile development processes. Consider this: Typical provisioning time for an enterprise-grade development environment can take, at a minimum, from several weeks to several months. Most Agile release cycles span four to six weeks at most. Not only does a four-week provisioning delay result in sub-optimal outcomes, but once created, these static environments are also difficult to change and cannot support the rapid iteration required for Agile development. Essentially, Agile development methodologies require Agile infrastructure for optimal efficiency.

Leading companies are changing the way their IT teams equip and support software development teams by integrating fast, dynamic, flexible, and easily shareable cloud-based environments that are available on-demand. They are changing IT architecture to incorporate cloud computing resources that enable Agile and empower software development teams with self-service. By integrating cloud-based services into the overall IT architecture strategy, software development teams are better enabled to create, change, and scale complex computing environments as often as needed. And at the same time, IT is able to retain the full visibility and control required for security and operational governance over these environments.

Enterprises that implement Agile IT architecture to support Agile development have seen provisioning of IT resources reduced from weeks to minutes, accelerating software release cycles dramatically. For example, Cushman & Wakefield, the world's largest real estate services firm, moved its application development and test environments to the cloud. In fewer than four months, Cushman & Wakefield development teams saw application release cycles shorten and become more efficient—and they doubled the number of projects they could complete in a given period of time.

**Below are five best practice guidelines that have served as crucial enablers of their success:**

1. Don't change the Agile development process to fit into legacy IT—change the IT strategy to better enable Agile software development.
2. Trust the developers. Empower them with self-service environments.
3. Expect rapid changes and fast iteration to be the new normal. Architect IT to be agile, configurable, scalable, and flexible.
4. Collaboration is at the heart of Agile development. Create an IT strategy where environments are easily sharable across disparate teams.
5. Agile IT should not eliminate security and governance requirements. Retain full visibility and control over your IT operations.

In this whitepaper, we will examine how to implement cloud-enabled software development by adopting an Agile IT strategy.

# Agile Development Defined

Most enterprise software development teams still use a traditional development life cycle model known as "waterfall." As Figure 1 shows, developers who use this model will spend a considerable amount of time defining requirements up front. Developers then focus on building the software according to the defined requirements. Once the software is ready for testing, the testing team analyzes payload against the specified requirements, eventually creating shippable software to the customer. This process can take anywhere from 6 to 12 months or longer to ship a single new release of the software application.

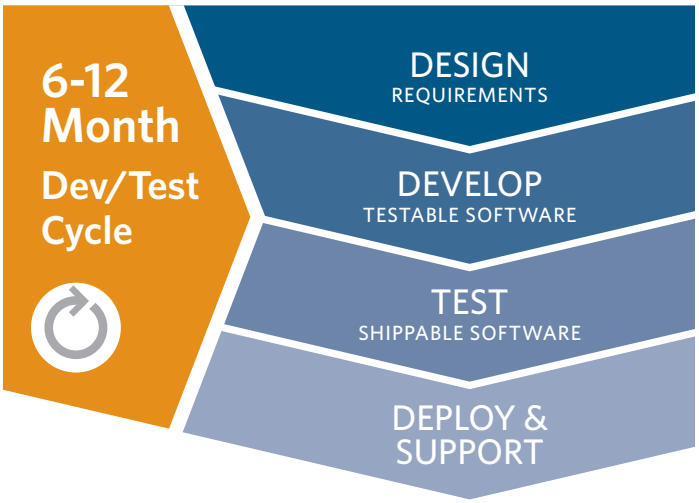


**Fig. 1: Waterfall development cycle**

From an IT perspective, the predictability of the waterfall development model is well-supported by existing IT environments. If development environments change, those changes can be better accommodated because time and resources are more readily available to set up new environments or alter existing ones.

However, the waterfall model has one huge drawback. It cannot adapt to rapidly changing business conditions and requirements. Development teams that use this model often find themselves rewriting requirements and requesting additional environment changes from IT. In the time it takes to complete this task, the business conditions may have changed once or multiple times, resulting in confusion and wasted development time. When this occurs, siloed IT and development teams inevitably become pitted against one another.

Frustrated with the delays and inadequacies of the waterfall model, a group of software industry veterans teamed up to examine the core process of software development, and after several years of experimenting with various techniques, they created the Agile development model with one singular goal: Deliver high quality software to customers faster and more efficiently.

Defined by the Agile Manifesto, the Agile development methodology subscribes to a core set of principles that instruct development teams to:

» Place the highest priority on satisfying the customer through early and continuous delivery of valuable software.

» Deliver working software frequently, within a few weeks to a few months, with a preference for a shorter lifecycle.

» Collaborate with line of business users daily, throughout the project.

» Build projects around motivated individuals, giving them the environment and support they need, and trusting them to get the job done.

» Use working software as the primary measure of progress. Sponsors, developers, and users should be able to maintain a constant pace indefinitely.

» Regard simplicity as absolutely essential.

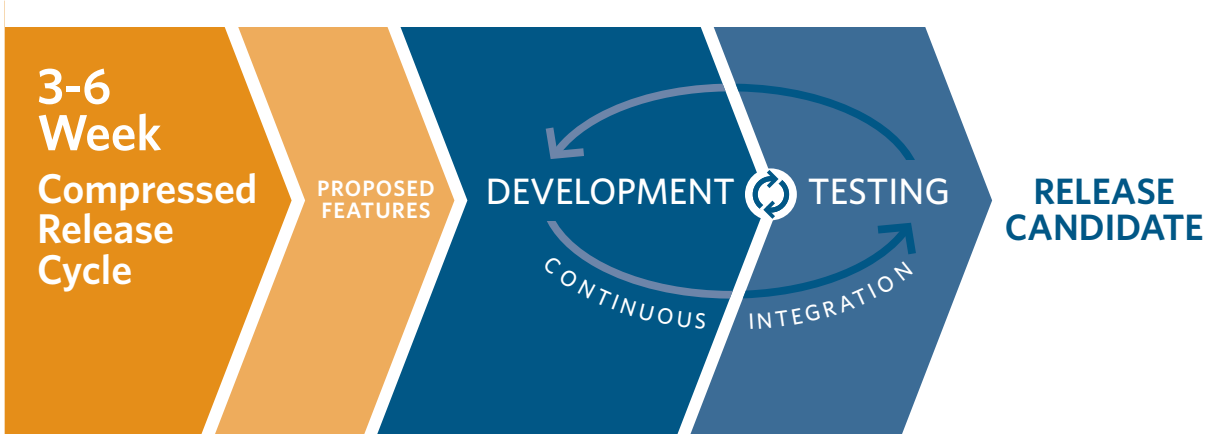


**Fig. 2: Agile software development life cycle**

As shown in Figure 2, the Agile model shifts from long planning, development, and testing cycles of 6 to 12 months to shorter 3 to 6 week release cycles. This methodology highlights the effectiveness of shipping software that solves near-term customer requirements.

# How IT Supports Agile Development and Testing Today

When developers adopt the Agile model, software development will undergo rapid iterations before being deployed to production. Agile development teams must have the flexibility to develop and test each iteration of code as they converge on the final version of product. Often times, developers must also test their code across multiple operating systems, network configurations, and browsers for each release. Most enterprise applications require a complex multi-tier, multi-network architecture that includes web servers, application servers, databases, firewalls, and load balancers. The ability to rapidly create, change, and test against these complex computing configurations is critical for Agile development speed and success. Unfortunately, most Agile teams do not have access to nimble, self-service environments that can support the high rate of change demanded by rapid iteration. Instead, progress is impeded by a variety of roadblocks inherent to legacy IT environments that are too rigid, costly, or static.

**Specific roadblocks may include:**

» Slow to provision/static development and testing environments – Developers and test engineers often must rely on IT teams for the ordering, set-up, and access to development and test environments. Once provisioned, these environments tend to be static.

» Underpowered infrastructure – The provisioning of environments typically involves older infrastructure that performs poorly and does not scale well. Often times the development and test lab is overloaded with too many overlapping projects.

» Lack of test coverage – In order to reproduce production scenarios or customer-specific issues, development and test teams need to be able to change operating systems, browsers, and databases, or infrastructure parameters such CPU, memory, disk size, and network configuration. Existing data center architectures built for predictable workloads lack the agility or configurability to support the dynamic requirements to cover the full test matrix with each pass.

» Inability to capture and reproduce complex software bugs – When developers and test engineers encounter complex bugs, they need to be able to save the memory, disk, and network settings across multiple machines and networks, so that the development team can properly diagnose the issue, while the QA team works in parallel on additional testing. Most development and test labs do not have the capacity or the flexibility to parallelize the capture of bugs with full environmental data and continue to execute on the full test matrix.

» » Limited sharing and collaboration – Most software development is team-oriented, and a lack of collaboration creates silos across teams and slows the release process. The challenge of collaboration and sharing is further compounded when the development, testing, and operations teams are geographically distributed with separate labs. Even more challenging is the case where constituents outside the company require access—examples include customers for user acceptance testing or contractors who perform certain test functions. IT teams using only in-house infrastructure often rely on scheduling specific shifts and resources for specific projects, and are typically unable to maintain the level of access or the pace of change required to support the multiple teams.

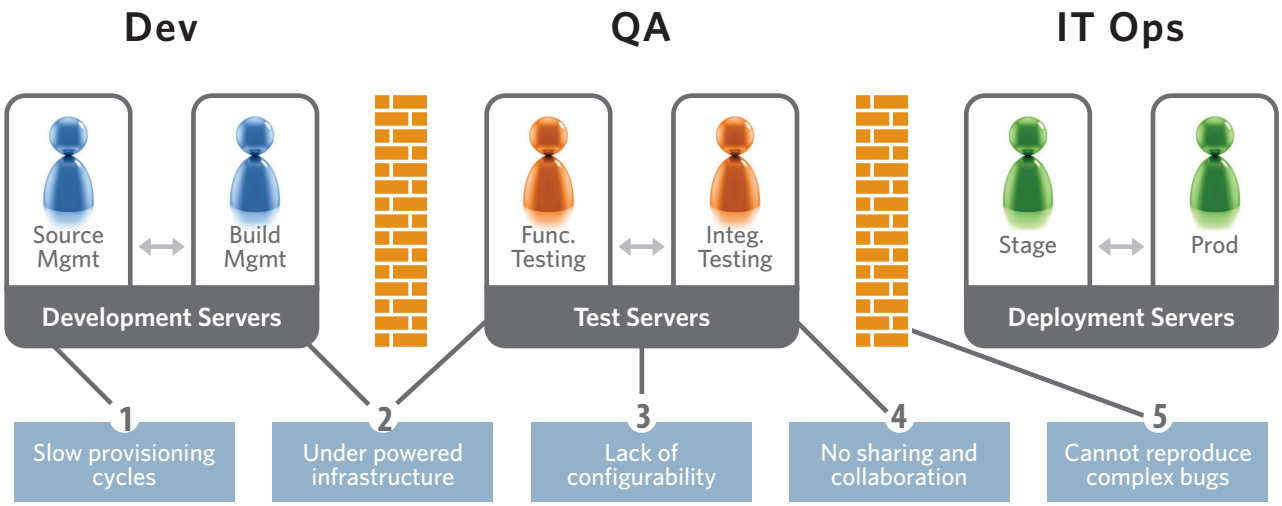


**Fig. 3: Challenges software development teams face with legacy IT**

In short, legacy IT environments are simply not designed to handle the demands of today's Agile software development cycles and complex application requirements.

Development teams are being pushed to create quality applications faster and at lower cost than ever before. Many are rightfully choosing the Agile model so they can focus on customer satisfaction, initiate rapid development, and build a culture of collaboration. But how can these teams succeed if they don't have the IT environment required to support their goals?

# New IT Requirements for Agile Development Teams

As discussed earlier, the goal of Agile development is to shift from a long and inflexible development process to a much shorter, more collaborative process directed toward shipping software more frequently.

To be most successful, Agile development teams will require the following from their development and test lab infrastructure:

» Self-service provisioning – Software development teams need to be able to create, change, deploy, copy, re-create, delete, and change development and test lab environments on demand, without IT assistance.

» On-demand scalability – Software development teams need to scale environments up and down easily.

» A library of virtual data center (VDC) templates – Software development teams need to create a consistent version of the current release stack, prior release stacks, and customer-specific variations as templates. Within a few minutes after a hot fix issue is reported, teams should be able to create a new environment that matches the appropriate release scenario to reproduce the issue, fix it, test it, and deploy the new code.

» Complex bug capture and reproduction – QA and support teams need to easily create, snap-shot, and clone complex, multi-tier environments when a complicated bug is identified. They should be able to capture and quickly recreate complex environments, including all memory and state information, so that the relevant development team resources can work to identify root cause and develop a fix, while the QA team moves forward with additional testing.

» Collaboration – Developers need to share copies of their lab environments with test engineers, other development engineers, and users. Teams should be able to organize their work in projects, invite specific project members to participate, and assign specific roles or access points to each participant based on roles (user, manager, database engineer, etc.) or status (employee, contractor, etc.).

Together, these requirements enable Agile software development teams to optimize the software development model for faster, more collaborative release cycles. When the development environment they use is as Agile as their development process, they can focus on specific customer problems, quickly develop solutions, iterate with customers, and ship higher quality software faster.

# Create Agile IT Using the Cloud

A typical procurement cycle requires 6 to 8 weeks to specify a development and test environment,; procure server, networking and storage hardware; and rack, configure, and test everything. That is a time-consuming, and costly process that does not match the demands of the Agile development cycle. Leveraging cloud computing resources can provide more convenient, affordable, and on-demand computing environments tailored to the needs of software development teams. By leveraging cloud computing resources as part of an overall IT strategy, enterprises can effectively extend their existing datacenters and manage the cloud as an extension of their existing environment.

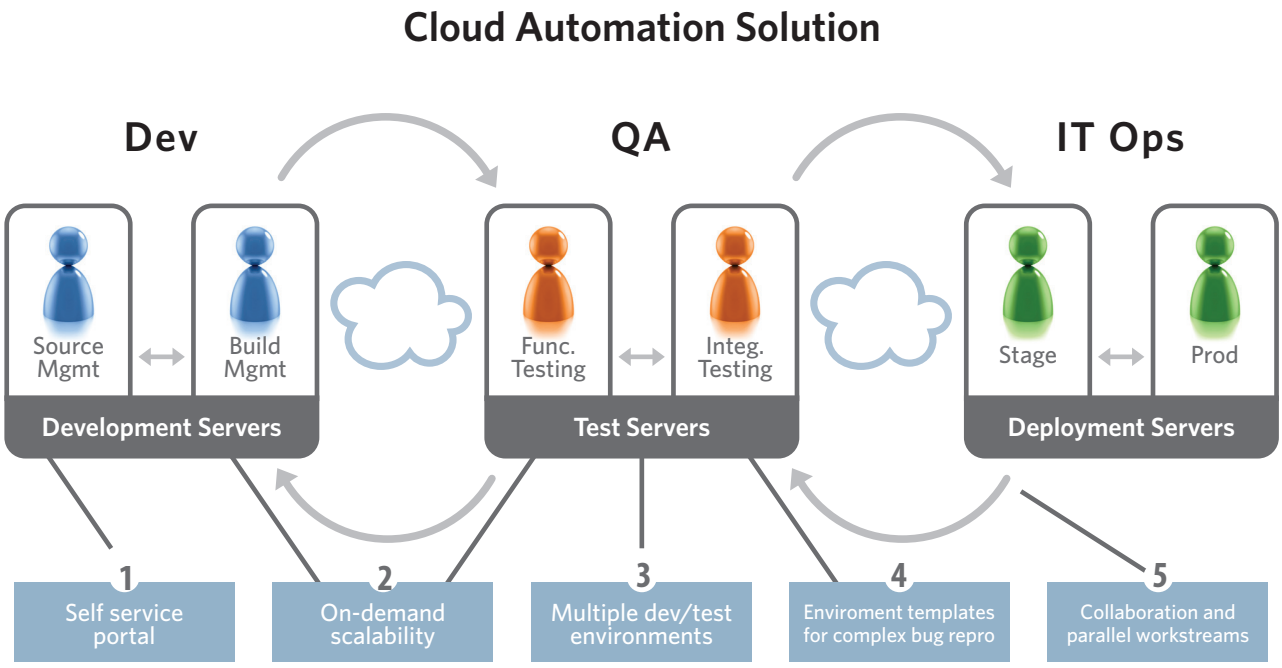


**Fig. 4: Cloud-enabled IT for development and test**

By utilizing cloud computing resources for development and test environments (rather than owning and maintaining hardware), IT can provide a higher degree of self-service for end users (developers and testers), and more configurability and scalability to support Agile requirements with much lower operational costs. Cloud-enabled solutions for development and test workloads will integrate the best characteristics of virtualization, cloud automation, software as a service (SaaS), and infrastructure as a service (IaaS) to provide a complete application lifecycle solution. This solution-centric approach enables:

» Developer and tester self-service – Developers and testers can create, replicate, change, or delete entire software development and test stacks and deploy application builds with just a few mouse clicks. A cloud-based solution can better enable teams to implement continuous integration, so they can quickly execute unit and functional tests and understand how their new code interacts with the rest of the application stack.

» Scalability and configurability – Developers and testers can create new release stacks on-demand, in a repeatable and dependable fashion, as they go through the peaks and troughs of release cycles.

» Broader test coverage – The cloud can enable developers and test engineers, virtually unlimited resources as compared to in-house physical lab infrastructure, which typically struggles to keep up as test matrices grow and release cycles compress. In the cloud, development teams can easily run multiple test passes in parallel to test various OS/DB/browser combinations, as the cloud enables VDC templates to scale up or down as needed. A rich library of VDC templates makes it easy to select from a broad range environments that are ready to be provisioned with a mouse click.

» Rapid bug capture and reproduction – In the cloud, test teams can snapshot complex bug environments (creating VDC templates), rapidly create clones of these templates, and continue to run tests unblocked by the saved snapshot in the development team queue. Using templates in a cloud-based environment enables testers to capture the whole environment without having to write down reproduction steps. Developers can then re-use those same environments to review the bug and quickly develop patches.

» Collaboration and parallel work streams – More than ever, application development teams are geographically dispersed, providing significant challenges to cross-team collaboration. With cloud-based solutions, developers can easily collaborate and share access to environments with other developers, testers, and offshore or contract resources by creating release-specific projects, and providing role-based access as appropriate.

# IT Requirements for Agile Cloud

Just as development and test teams demand that cloud infrastructure support their Agile development needs, IT professionals also require that cloud-based labs meet requirements as they relate to managing budget and user access. Specifically, IT's need for full visibility and control of cloud environments requires that they can:

» Set up development and test environment templates that are IT policy compliant.

» Create users, roles, access control lists, and set permissions.

» Establish a hybrid cloud architecture to securely connect the cloud environment to existing data center infrastructure.

» Assign group, project, and individual-level quotas for machines, storage, and networks.

» Track usage by month, by user, by project, and implement chargebacks if needed.

» Audit and ensure compliance policies are followed.

IT teams that adopt a cloud-enabled Agile IT solution can:

» Increase business agility for development teams.

» Reduce time to market for new applications.

» Ensure development ships better software faster.

» Boost productivity across development/test and IT teams.

» Lower fixed and variable costs.

Agile IT empowers development teams to achieve the full potential of the Agile model. In addition, IT will be able to retain full visibility and control over cloud environments and reduce operating costs.

# Five Best Practices for Creating Cloud-Enabled Software Development

Modern enterprise organizations are leveraging cloud computing to enable IT to be Agile and to power Agile software development teams. Companies that have successfully implemented cloud-enabled software development have subscribed to the following five best practices when creating Agile infrastructure:

1. Don't change the Agile process to fit legacy infrastructure; change your IT strategy to be more Agile. The very essence of the Agile model is trust and delegation, and yet some IT organizations still struggle to operate with these principles in mind. They claim to support the Agile development model, but require developers to change their model to fit the existing IT processes. Successful IT organizations are flexible and work with software development teams to create an IT cloud strategy that is self-service oriented, while still providing visibility and control for IT.

2. Empower end users with self-service environments. Enable your developers and testers by creating VDC templates specific to each development project, and providing IT services that developers can consume easily and without intervention.

3. Expect rapid changes and fast iteration to be the new normal. Be ready to architect your cloud infrastructure to be more configurable, scalable, and flexible. An Agile development model is inherently fast paced, so be willing to accommodate change and quickly adapt to what works and what does not. Expect rapid iteration and design your cloud implementation accordingly.

4. Collaboration is at the heart of Agile development. Customers, line-of-business users, QA engineers, contractors, and support professionals should be able to collaborate during multiple phases of the development cycle. All of these stakeholders are expected to operate on the application based on the specific roles they play on the team. If the environment cannot be easily replicated and shared across teams, then the developers and testers will struggle to get the full benefits of Agile.

5. Maintain full visibility and control over IT operations. Implementing Agile cloud infrastructure does not eliminate security and governance needs. IT organizations need to set security policies and enforce them through granular access control. They need to have full visibility into quota usage, resource management, and compliance.

# Conclusion

Extending your existing IT architecture to encompass a cloud-enabled development strategy will ultimately serve your Agile development team better. The process does not have to be difficult or challenging, and in most cases, complex computing environments can be created in the cloud in just a few minutes or hours versus days or weeks. Look for enterprise-grade cloud computing service providers that can deliver on the following five key capabilities to enable your applications development teams:

1. Intuitive self-service
2. Fast productivity
3. Flexible, complex computing environments
4. Collaborative platforms for teams
5. Full visibility and control for IT

Although it is possible to continue using on-premises infrastructure, your Agile development process will be much more effective when your IT infrastructure and service delivery model is Agile. Given the potential consequences of operating with dated hardware, poor collaboration, and slow provisioning of IT resources, organizations can increase business agility by embracing cloud computing for software development and testing.

# About Skytap

Skytap provides Environments-as-a-Service to the enterprise. Our solution removes the inefficiencies and constraints that companies have within their software development and test lifecycles. As a result, our customers release better software faster.

Today's enterprise is challenged to continuously deliver new customer-facing applications, while overcoming increasing change and complexity in their IT infrastructures. Our customers use Skytap to manage, import, deploy and decommission on-demand environments that contain everything needed to accelerate the software lifecycle, without unnecessary costs and project delays due to manual configuration and dependencies.

Enterprise IT organizations maintain full visibility and cost control, while allowing dev and test teams to self-provision labs, and copy and share complex environments across global cloud regions with ease for a lasting boost in agility.

Customers can import existing virtualized applications or build new applications in the cloud. Skytap can be easily accessed through any modern web browser, REST-based API, Command Line Interface (CLI), or ALM tool (Jenkins, Visual Studio TFS, etc.).

Skytap customers have a choice of infrastructure. Customers can run complex computing environments on Skytap's native ESX-based infrastructure, or leverage our services atop leading cloud infrastructures such as AWS and Softlayer.

**Skytap, Inc.**

719 2nd Ave, Suite 300
Seattle, WA 98104 USA

Toll Free: +1-888-SKY-TAP8 (1-888-759-8278)

Web: **www.skytap.com**