MACSTADIUM | 2021 WHITE PAPER

# Transitioning iOS Continuous Integration from On-Premises to Cloud

# Transitioning iOS CI from On-Prem to Cloud

Effective continuous integration (CI) is essential to successfully building modern iOS applications. Being able to iterate on application releases quickly and cleanly is central to building and maintaining an active and dedicated user base. Moreover, successful CI processes are vital to maintaining developer satisfaction and, ultimately, driving developer retention, which is also fundamental to lasting success in this highly competitive marketplace.

In our experience at MacStadium, we find that iOS development teams generally recognize the inherent value in CI quite early. So early, in fact, that they are often still emerging as players in the field. Because of this, teams generally begin their CI for iOS while still working under one roof and while the development team is still relatively small. A small, localized development team makes small-scale, localized CI processes running on bare metal machines a viable and cost-effective solution. This is particularly true in the case of iOS CI, as associated build and test processes need to be executed on certified Apple hardware, which can be cost-prohibitive to purchase and maintain at scale.

However, in such a high-growth market, successful teams inevitably evolve, and CI solutions need to grow and evolve with the team to retain their inherent value. When these factors do not evolve in tandem, you run the risk of losing trust in the integrity of your CI on the part of engineers, and your hidden costs of continued maintenance can skyrocket.

## CONTENTS:


**CI and Application Release Cadence** 3

**CI and Developer Satisfaction** 3

**Hidden Costs of Outgrowing Your CI Solution** 4

- Inconsistent Build Environments 4
- Brittle Pipelines and Required Maintenance 4
- Expanding Build Queues 4

**Cloud-based ClaaS Solutions** 5

**MacStadium for Scaling iOS CI** 5

**Adding Virtualization** 6

- Orka 6
- Anka 6
- VMware 6

**Conclusion** 7

**References** 7


## KEY TAKEAWAYS:

- An effective CI pipeline is essential for driving a frequent release schedule, as well as for maintaining developer satisfaction
- An outgrown CI system can lead to increased build queues, maintenance windows, and troubleshooting time
- ClaaS can be an easy-to-deploy solution for teams not yet ready to build a custom solution
- MacStadium offers diverse and customizable CI solutions built on genuine Apple hardware that can be virtualized for additional efficiency

# CI and Application Release Cadence

When development teams employ effective CI, they release software an average of more than twice as often as those that do not [5]. This increase in the rate of application releases is essential to maintaining a competitive posture in such a competitive landscape [4]. User bases are built and maintained, in large part, through the regular and error-free release of upgrades and new features within an application.

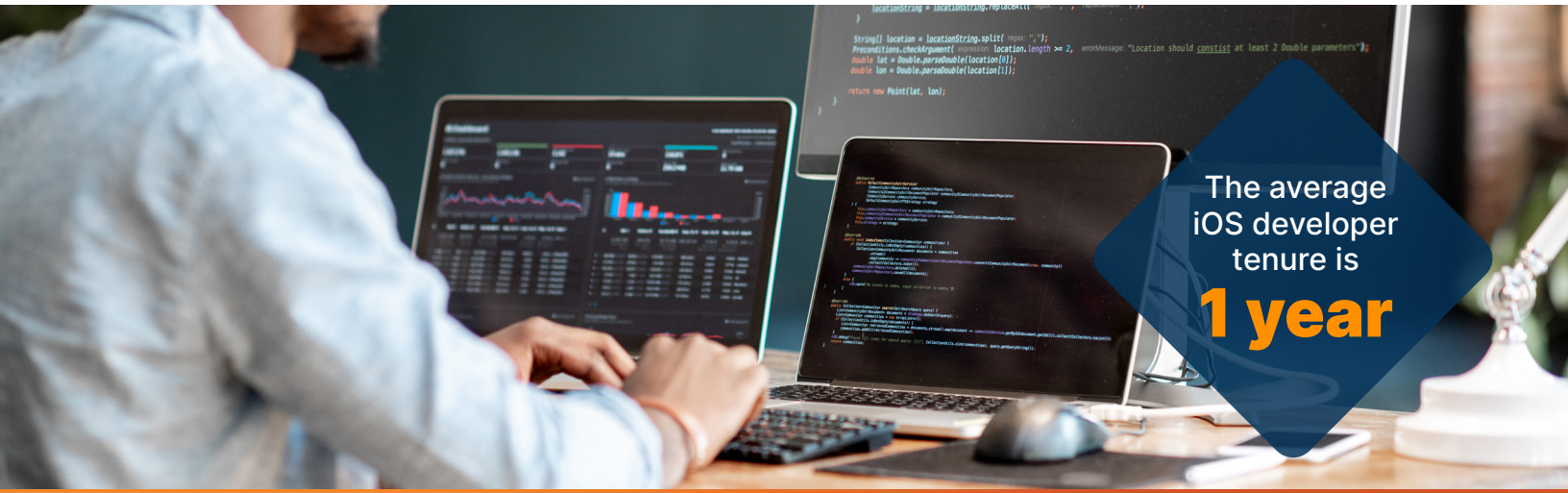
However, effective CI will not remain effective indefinitely. It will likely need to evolve with the development team it is serving in order to maintain this increase in application release cadence [2,3]. In particular, maintaining effective CI elements such as reasonable build queues, consistent pipeline performance, and secure access for the full development team [2,3] will likely require fundamental changes to what was an effective CI pipeline for a given team at an earlier stage of its growth.

# CI and Developer Satisfaction

Developers thrive on seeing their work out in the wild. Grueling development sprints are quickly forgotten in the face of a new application feature that a developer can take a sense of pride in having released. However, when the same grueling development sprint is met with ineffective, or unreliable CI, this can lead to a stalling out in the release cycle [2,3,4] – potentially to the point where the newly developed feature is no longer in scope for a given project by the time that it is actually deemed ready for release [4].

In addition to the obvious waste of resources in the form of a highly paid development team's time, there is also the very real possibility that the perceived costs of development sprints on the part of developers themselves can begin to outweigh the perceived benefits of inherently challenging work.

If left unchecked, this cycle can lead down a path of eroding developer satisfaction, and can ultimately jeopardize a team's ability to retain developers [4] – an undeniably essential resource for lasting success in this competitive market.



# Hidden Costs of Outgrowing Your CI Solution

## Inconsistent Build Environments

Build environment consistency is essential to achieving a concrete baseline for tests and other CI processes to execute against [2,3]. When static build environments, those that are left up indefinitely for multiple workflow runs, are not returned to their original, clean state between workflow runs, there is a risk of environment drift, which can result in inconsistent or failing tests.

As a product matures and its CI processes become increasingly complex, it becomes equally complex to ensure that environment cleanup scripts run between workflows are fully effective in regaining an original, clean state. This increasingly complex workload of maintaining the environment cleanup scripts generally falls on experienced developers, and as such, it constitutes a significant financial and temporal cost [2,3].

## Brittle Pipelines and Required Maintenance

Perhaps the greatest immediate financial cost incurred by a development team that has outgrown, but is still employing, an early iteration on a CI pipeline is the hours spent on maintaining the pipeline by high-value engineers who would otherwise be engaged in directly creating value for the company [2,3].

Engineers can easily fall into the trap of committing hours on end to maintaining the build server itself, as well as troubleshooting networking difficulties that arise from fragile workarounds deployed to extend the life of the CI pipeline beyond its originally scoped use case, such as brittle SSH connections to the build environment from remote developers' local development environments [3].

## Expanding Build Queues

Another hidden cost with the potential to grow exponentially is the downtime that developers experience while they wait for their CI jobs to execute [2,3]. New code cannot be committed to the project until the requisite tests have passed, which can take an extended period of time, to begin with. When the CI pipeline is tied up by other commits being processed, this means that developers will have to wait for not only their own CI job to execute, but also for any jobs running ahead of theirs in line, because the CI pipeline will need to wait for an available build server on which to execute the CI job.

In addition to the time that is lost waiting for CI jobs to execute, the longer a developer has to wait for his or her tests to run so that the newly written code can be committed to the project, the further he or she will move away from the specific logic that he or she is waiting to return to. Context switching as a result of extended build queues can thus further multiply the time that highly paid developers require to reengage with a given portion of the codebase [6].

### 4 Signs You've Outgrown Your iOS CI Pipeline

Build queues are steadily increasing

Networking failures are frequent

Build server maintenance time is increasing

Developers' confidence in existing CI is eroding

# Cloud-based ClaaS Solutions

Continuous integration as a service (ClaaS) is an attractive option for many teams as a transition to cloud-based CI because teams with limited in-house DevOps expertise can very easily mitigate many of the growing pains that arise with early iterations of on-premises CI described above.

These services are effectively plug-and-play in that they offer a pre-built mechanism to run ephemeral CI for iOS, which can be very attractive to teams that are looking for a quick, easy-to-deploy solution or are not yet ready to build a custom solution.

The convenience that these services offer can, however, become expensive as teams and their products grow. When there are still relatively few developers actively committing to a project, and the project itself is still relatively small, this cost can be reasonable, as the time per build and the total number of builds executed are relatively low. However, as the development team and the product itself grow, the total build time required, and the associated cost of these builds can grow exponentially.

As teams mature, they often grow not only in size but also in expertise. More in-house expertise, combined with the parallel increase in CI costs, often leads teams to explore custom iOS CI solutions built on cloud-based infrastructure as a service (IaaS) such as MacStadium provides.

# MacStadium for Scaling iOS CI

MacStadium's private clouds or dedicated Mac hardware is the best choice for transitioning your iOS CI processes from an early, localized iteration or an intermediate ClaaS-driven iteration into a mature, cloud-based operation. MacStadium's diverse set of managed macOS compute solutions will allow you to refocus on your core objective – building best-in-class iOS applications. MacStadium significantly reduces the technical overhead involved in scaling secure and reliable iOS CI systems by managing truly diverse, genuine Apple infrastructure that you can run as bare metal, or with one of our three distinct virtualization solutions.

MacStadium maintains a global footprint and 24/7 access to remote hands and Apple Mac networking experts should any problems with your build environment arise. MacStadium is trusted by thousands of DevOps teams around the world and can help you achieve a highly performant and scalable iOS CI solution tailored to your specific team and use case.

# Adding Virtualization

While bare metal builds are ostensibly faster because they can leverage the full compute capacity of the build machine [2,3], virtualization can save a tremendous amount of time and error-ridden effort returning a static build environment back to its original, clean state [2,3]. This time savings, coupled with the avoidance of eroding confidence in the quality of the CI processes generally on the part of engineers lead teams operating at scale to often opt for virtualized solutions for iOS CI [2,3].

MacStadium is the industry leader in macOS virtualization solutions. We offer three distinct macOS virtualization solutions Orka, VMware, and Anka. Because iOS CI needs to be executed on genuine Apple hardware, this positions MacStadium as the premier option for executing iOS CI at scale.

If you'd like to add macOS virtualization to your build pipeline, MacStadium sales engineers are available to help guide you through the options and determine which will be the best fit for your team.


### Orka

- Highly scalable solution optimized for ephemeral VMs and resource management across concurrent builds
- Developer-focused platform built on Kubernetes technology with dev-first user interfaces
- Access to standard tools like KubeCTL, KubeDashboard, Autoscaling, etc.


### Anka

- Get started with macOS virtualization quickly on the latest macOS release
- Built specifically for macOS, running natively on Mac for optimized performance
- Ability for your developers to pull VM images from a central registry, then work with VMs on their local machines


### VMware

- Industry-standard virtualization solution for enterprise environments
- Good if you have VMware deployed elsewhere in your organization
- Need security and compliance requirements not met by the containerized deploy clusters on Anka or Orka

# Conclusion

Effective CI for iOS is essential for driving an application release schedule that is conducive to growing and retaining an active user base. Moreover, it is essential to maintaining developer satisfaction and is thereby a primary driver of developer retention in an industry that is marked by notably high turnover rates.

CI systems that were once effective may no longer be. Specifically, as development teams grow and evolve, the CI systems that serve them must also grow and evolve to retain their effectiveness. Indicators that a development team has outgrown its CI system include growing build queues, increasing time spent on maintaining build servers, and increasing time troubleshooting brittle networking workarounds.

MacStadium is a proven solution for evolving teams that find themselves in need of a mature, cloud-based iOS CI solution. With our industry-leading security posture and macOS virtualization solutions, you can iterate on your CI pipeline with confidence that your CI processes will be highly performant and secure while freeing your high-value engineers to focus on delivering value rather than maintaining a system that you have simply outgrown.

**Learn more:**


www.macstadium.com


docs.macstadium.com


blog.macstadium.com


sales@macstadium.com

# References


[1] Makam, Vasanth. (2020) Continuous Integration on Cloud Versus on Premise: A Review of Integration Tools. 10-14. 10.5923/j.ac.20201001.02.

[2] Niderberg, Alex. (2019) 4 Lessons From Scaling iOS CI/CD: From a Snowflake Build Machine to Empowering Hundreds of Mobile Engineers to Deliver Delightful Mobile Experiences.

[3] Khai Koh, Su. (2020) Our iOS CI/CD Journey at Nextdoor.

[4] Buchanan, Chrissie and Chia, William. (2019) The Business Impact of CI/CD.

[5] Hilton, Michael et al. (2016) Usage, Costs, and Benefits of Continuous Integration in Open-Source Projects. Oregon State University.

[6] Tregubov, Alexey et al. (2017) Impact of task switching and work interruptions on software development processes. 134-138. 10.1145/3084100.3084116.