
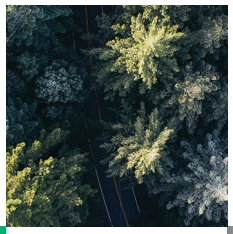
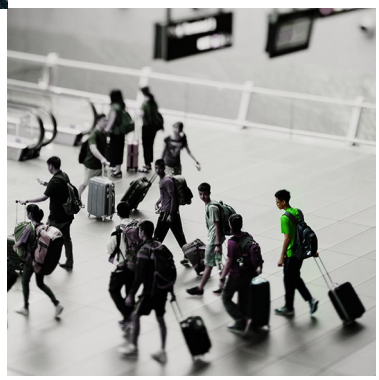
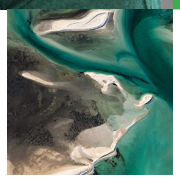

A NetApp SolidFire Insight

Five Continuous Deployment Lessons You DON'T Want to Learn on Your Own

Which of these two situations below is worse?
It's 2 a.m. Your crying, hungry baby beckons and your spouse says, "We're out of formula."

Or:

It's 2 a.m. The phone rings and the voice on the other end wails, "The system is down."

The answer is easy.
At least you can buy more formula from the 24-hour store.

Development success today is synonymous with upgrades and refinements that proceed apace, without performance interruptions and downtime for the rest of the organization. This continuum of IT service improvements is a requirement of today's highly dynamic business environment, not just a nicety to keep the CIO off your back. You simply cannot continuously refine, upgrade and deploy while taking the system down to do so.

This article was originally published on TechTarget (searchsolidstatestorage.techtarget.com) and authored by Bill Laberis, Partner, Gillin & Laberis – Original source

NetApp®

Here are five deployment lessons you cannot afford to learn on the fly in an era when downtime can have such disastrous consequences:

Lesson 1

Ensure that upgrades to improve development are non-disruptive.

This can be a shared responsibility, in part. When working with key vendor partners on upgrades using their technology, make non-disruption of business as usual a requirement that both of you have to consider a priority. Having an underlying storage platform designed from the outset to be self-healing and to upgrade without downtime will further aid the cause of non-disruption. This means the system overall will continue to function even if multiple concurrent nodes fail. Adding this type of functionality to legacy architecture "after the fact" is difficult and often impossible.

Endicia, a leading supplier of electronic postage software solutions, leveraged an all-flash array storage solution to get exactly this: a next-gen storage solution that virtually eliminated downtime and allowed the company to add additional capacity-on-demand without any service disruptions.[1]

Lesson 2

Develop a strategy to scale seamlessly and without adding IT complexity.

Stripped down to the essential, scaling typically means adding more pieces to the infrastructure, which often means adding more overhead and complexity. Successful SaaS delivery requires continuous scale, but not continuous overhead and complexity. Growth is not only expected, it is required for SaaS and PaaS offerings to be successful. SaaS and PaaS offerings need to be stateless, scaling readily to deal with spikes and potential disruptions. So what do you do? Users often struggle to manage discrete pools of storage for specific instances, workloads and apps, which adds successive layers of complexity as they grow. Does this sound too good to be true: non-disruptive scaling storage platform capable of supporting all workloads, across thousands of instances with guaranteed quality of service (QoS) for each application, rapid space efficient cloning and a unified API? Check out the reference to Endicia in Lesson 1 above and see this strategy in action.

Lesson 3

Be programmatic by having developers write code that doesn't need constant babysitting and tweaking.

Tired of managing the ever-sprawling infrastructure with time consuming, costly manual processes? Then familiarize yourself with Infrastructure as Code (IAC), a.k.a. programmable infrastructure. Programmability involves using higher-level descriptive language to automate so much of the manual tedium in operations and management. As technical professionals developing and delivering software innovation, we are constantly adopting and utilizing new technologies. Developers are no different and are eager to learn and leverage new technology.

The question to ask at the outset is: What will make incorporating new technology the most ergonomic for the developer? Programmable infrastructure frees up developers to do what they really like, namely building new and improved IT services and solutions. As explained by noted tech writer Margaret Rouse, instead of relying on sys admins to manage operations in a DevOps environment, developers instead can write an IAC and essentially roll their own quality assurance or experimental deployment, saving steps, money and time.[2]

Lesson 4

Avoid the downtime-provoking forklift upgrades as you adapt to new technologies.

This lesson comes with a healthy dose of scale-out, which as in Lesson 2 relates to working closely with key vendors. From storage media and network connectivity to programming languages and orchestration stacks, there is a regular and continuous stream of new innovations that SaaS and PaaS providers can leverage to deliver improved customer experiences. With this continuous stream, we know that the latest new thing won't be the last new thing. As new technologies appear, you need to be able to incorporate and take advantage of them quickly in a manner non-disruptive to your existing service delivery. As in Lesson 3, the capability to non-disruptively take advantage of new technologies as they emerge is difficult and often impossible to bolt onto a legacy storage architecture.

Lesson 5

Make sure the new tools you introduce are the ones developers are actually using.

This lesson of course implies that you have the infrastructure in place so that new tools can seamlessly plug into it. What tools and solutions? To enhance operations automation, consider tools like Puppet, Chef, SaltStack and Ansible.

1 "Video Case Study - Encidia," SolidFire
2 "Infrastructure as Code definition," SearchCloudComputing