Nitro Moves Desktop Apps to the Cloud with Typesafe Reactive Platform

Nitro's VP of Technology outlines the importance of Play, Akka and Scala to the company's product delivery evolution during recent years of explosive growth

About Nitro

Nitro is changing the way the world works with documents. From the desktop to the cloud, the company's solutions make it easy to create, edit, share, sign and collaborate—online or offline. Learn more at: https://www.gonitro.com/

Nitro's Journey to the Cloud

When Tihomir Bajic joined Nitro in 2011 as VP of Product Engineering, the company was embarking on its vision to build Nitro Cloud, and his responsibility was to evolve its popular software for document productivity and workflows from desktop-only to a SaaS distribution model.

Nitro's migration to the cloud was no small challenge - the company already had millions of users and a massive code base of C++ for desktop apps for document processing and workflows. Nitro's desktop products were supported by .NET web services running in a public cloud. At the time, it was uncertain exactly what Nitro Cloud would look like as a product platform, but Bajic understood that what was most important was that platform decisions could support the journey from proof of concept, to prototyping and beyond. Along the way, Typesafe technologies enabled Nitro's journey to Reactive.

Play Helps Nitro's Early Prototyping

Prior to joining Nitro, Bajic had seen Play Framework being used by a friend to bootstrap a production-ready application in only a couple of weeks. That startup, AppHero, was an Apple Store recommendation engine based on social and user preferences (acquired in 2013 by mobile app publishing firm Fuse Powered). In addition, through sponsoring a Play Framework hackathon in December 2011, he became convinced it was a framework others could pick up over a single weekend.

Based on his exposure to using Play for rapid prototyping, Bajic selected the framework in late 2011 as a development framework. Initially his team used Java Play 1.2 hosted on Heroku, and Git Push Remote for code deployment - for what he called "a very cheap but effective version of continuous integration." Nitro found it exceptionally easy to write code

with Play and show working versions to internal and external users - and the process of iterating on his product went very smoothly.

Akka Helps Nitro Get Asynchronous in the Cloud

As Bajic and his team got further along in the migration from desktop to cloud, the team recognized how unpredictable and compute intense their users' document management jobs could be. Sometimes doing something as simple as converting a 1-page resume from PDF to Word - depending on the doc structure - can require a great deal of computation. Even knowing the size of the file or number of pages in advance, it was impossible to predict how long certain operations would take in the cloud.

Bajic realized that the team needed to build an asynchronous platform for document processing, and opted for a combination of Akka and RabbitMQ. Nitro adopted Akka because they liked the Actor model, and Akka's message-driven architecture.

In the process of leveraging Akka Actors to achieve asynchronicity in the cloud, Nitro saw an opportunity to stop using Heroku/AWS (no longer scaling compute cost efficiently) and build its own co-located document processing platform. Nitro Cloud's vision became more ambitious as the company moved towards customized networking, hardware and middleware to handle compute-intensive document processing, where Heroku still handled the web application layer, but all of the heavy lifting in document processing was occurring on the new Akka platform running in a collocated data center.

On Nitro Cloud, documents are atomic objects, and Actors can exact actions on those documents - such as changing their state or forwarding their revisions. Everything is designed to be highly asynchronous and message-driven. And despite all of the many transformations that can be imparted on the documents themselves, the relationships between the Actors and the documents are simple and scalable, and any failure is properly isolated and easy to manage.

Nitro Starts Replacing .Net With Play

As Nitro continued its push towards the cloud, and continued to deconstruct its monolithic C++ applications into cloud services, the company also started to take a hard look at existing assets running on .Net. A number of Nitro's customer facing web services, its corporate web site, and a giant master record database were running in a sprawling .Net monolith.

The .Net platform was proving unscalable, due to the size of the code base (already six years old) and how difficult it was to release to production. It was a blocker to hiring developer talent - on such a large code base on .Net, just compiling and seeing changes locally took a long time, and amounted for a "huge turn off" according to Bajic.

As Nitro moved towards a "web services first" attitude, it aggressively replaced .Net with Play, to increase its development cycles and release velocity. In just over a year (by start of 2013), Nitro was off of .Net completely.

Nitro's Reactive Journey, Present Through Future

Nitro has continued to scale, and continued to add new applications on top of its Cloud platform. The company's first app was Nitro Cloud. Account management portals and Salesforce integration came next, and along with it the requirement for a centralized Nitro platform. As more applications have been built on top of the Nitro platform, complexity has increased.

In late 2013, Bajic and his team were focused on how they were going to go "10x" - how they were going to completely eliminate perceived downtime, increase responsiveness for true web scale document processing and continue to evolve. Ultimately the goal for Nitro is for the platform to be available to third party developers to build smart document workflows on top of its platform, so this continuous performance evolution is key to the business strategy.

"Right around the time we were pondering these questions, we came across the Reactive Manifesto," said Bajic. "And it was like someone was telling us exactly what

we'd gone through over the last two years - importance of being message-driven, how responsiveness is the number one customer expectation, need to make your product more elastic and resilient in face of failure. It all resonated very highly for us."

Shortly after, Nitro adopted Scala (it had been running Play on Java previously). The company has favored Scala's strengths with asynchronous business logic, its expressional simplicity, its ability to help them structure and compose their code functionally in pieces that are easy to reason on their own.

Summary of Benefits

- ***Increased responsiveness and release frequency*** **–** The Typesafe Reactive Platform has helped Nitro deliver increasingly better user experience. Legacy .Net web services latency decreased by a factor of 10 on average. The Typesafe Platform has enabled service de-composition and also sped up the release process several times. This enables Nitro to release frequent (sometimes daily) updates for selected services only.

- ***Attracting bar-raising talent*** **–** Practicing the Reactive Manifesto principles on the Typesafe Platform has introduced Nitro to an amazing engineering talent community. Together with Nitro's business opportunities this has helped us attract some of the world's best engineering talent.

- ***Maximized productivity*** **–** "In the zone" productivity is a mindset. This mindset can be supported by a great toolkit and community like Typesafe's. Rapid, iterative development with immediate results, paired with Scala and Platform's compile-time safety harnesses, allow our engineers to have productive fun while coding.

*The **Typesafe Reactive Platform** is a modern software platform that makes it easy for developers to build scalable software applications. It combines **Play Framework**, **Akka** runtime, the **Scala** programming language, and robust developer tools in a simple package that integrates seamlessly with existing Java infrastructure. Commercial support and maintenance is available for the Typesafe Reactive Platform through the **Typesafe Subscription.***

info@typesafe.com