Building a new product on Typesafe

Building a production-ready social networking backend in three months with three developers is a daunting task. Learn how the Conspire team did just that with the Typesafe Reactive Platform.

About Conspire

Conspire is a TechStars company. They analyze email data to give users detailed analytics about their email network and to understand the strength of connections between people. With this understanding, they maintain an always-up-to-date, weighted network of connections without any work on the part of users. When a user needs to reach a person or company, Conspire finds the strongest path of connections in the user's extended network.

Paul McReynolds and Alex Devkar co-founded the company in the San Francisco Bay Area in 2012. Ryan Tanner, a highly experienced software engineer, was their first employee. After spending a short time in Texas for TechStars Cloud 2013, they set up permanent headquarters in downtown Boulder, Colorado.

The Business Problem

When Conspire graduated from TechStars, they started planning for their revamped and customer-facing product. Conspire's original backend was a multi-threaded Java application with a traditional concurrency model. The code was littered with semaphores, locks and database transactions. As a prototype, it was adequate, but it was already too complicated, and it couldn't serve as the foundation for everything that was to come. This meant a *lot* of new code; while the old codebase was functional and got the job done, the team wasn't particularly proud of it, and they were definitely not sad to see it go.

Choosing a Solution

During the prototyping phase, the team had had some great success with Typesafe's Akka for building parts of the solution—most notably a pre-caching service to solve some serious performance problems. Akka is a toolkit and runtime for building highly concurrent, distributed, and fault tolerant event-driven

Revision 1.0

applications on the JVM. The team was excited about some of Akka's upcoming features and, based on their experience thus far, decided to use it as the basis of their new backend.

Akka played well to the tenets the team was looking for in their product:

- Scalability
- Resiliency
- Simplicity

The programming language to use, however, was less certain. Ryan was an accomplished Scala developer with two years of experience under his belt, but the rest of the team were Java developers and had no such foundation. Since Akka—and the other components of the Typesafe Reactive Platform—can be used from Java just was easily as Scala, the team decided it would be prudent to begin with Java.

Moving to Scala

While choosing Java presented a shorter learning curve for the team, Ryan continued to do some development in Scala. By implementing parts of the backend in Scala, Ryan could clearly demonstrate its benefits, which put Scala in perspective and showed the team how it could help them write better code. While doing this, Ryan continued to pique his teammates' curiosity with links and articles on the benefits of Scala, such as Jonas Bonér's excellent talk on "Going Reactive" and Twitter's Scala School. The simplicity of writing type-safe, asynchronous code in Scala is a huge benefit when building a backend such as Conspire's, and through Ryan's diligent efforts, the choice to go with Java was reversed and Scala prevailed.

Paul and Alex were reluctant but Paul gave Scala a shot. His first time writing Scala was building out the first pass of a new analytics service. In one sitting, he went from no prior Scala experience to an analytics service that worked correctly the first time it compiled.

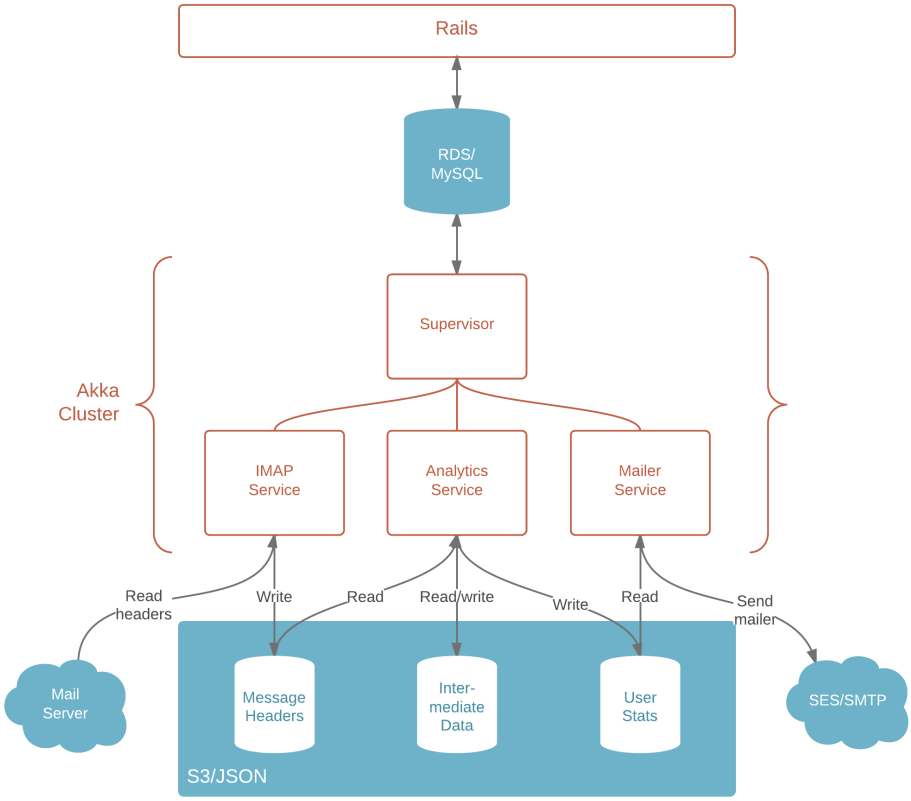
Ryan Tanner, Software Engineer, Conspire.

The Architecture

The architecture that the Conspire team chose enabled them to completely decouple the back-end from the front-end. Not only did this design simplify the number of code paths through the back-end, but also it could be taken down at will (or fail) without impacting the end-user experience. This was particularly important because nobody on the team had production experience with this new platform.

The team divided the backend into three siloed services—IMAP processing, analytics and mailing—all of which are overseen by a supervisor. Each service is designed to be independent of the others, communicating only via the supervisor. Reducing the cohesion between services reduces concerns about scalability and resilience. When cluster nodes can ignore the status of other nodes, they don't have to care when those other nodes fail, thereby truly embracing Akka's "Let it Crash" mantra.

Conspire's architecture looked something like this:



The team took a "KISS" approach to persistence: convert state to JSON and utilize Amazon S3 for storage.

Given our use case, we don't have any need to query individual pieces of data for users or across multiple users. We load all of a user's data into memory, process it and write the results back out. For a given user we easily end up with 50-100MB of data and Amazon S3 gives us a cheap, fast and reliable storage service without having to worry about scalability.

Ryan Tanner, Software Engineer, Conspire

With this architecture, Akka brings fault-tolerance and resilience to the table—any given node can fail without taking down the entire backend. Scaling out is as simple as spinning up a new Amazon EC2 instance and having it join the cluster; Akka simply starts using it as a worker node for whatever service it's registered with.

Moving Forward

The team built out the entire backend over a three-month period and is very happy with how things turned out, even though they had some frantic moments when they were afraid that Akka wasn't going to live up

to its promises. Luckily, being able to reach out to the Akka Engineering Team for support helped them through the tough times.

Akka takes care of the heavy lifting of our dual requirements for fault tolerance and elastic clustering.

Read Ryan Tanner's excellent series of blog postings on this application here.

*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

Revision 1.0