Adam McCullough

Software Engineer with a systems background, passionate about Functional Programming, correctness, and reliability.

Experience

Facebook, Menlo Park, CA— *Production Engineer*

March 2018-July 2019

Worked as a Production Engineer embedded in the Feed And Stories Team. FAST was responsible for maintaining the Facebook News Feed, which was the centerpiece of the Facebook web experience, both in terms of monthly active users, and, consequently, revenue. As such, it was a very large team (1,500 people during intern season), with a comparatively small team of PEs (7).

PEs oversaw both operational things, such as server infrastructure and capacity management, as well as monitoring and deploying new builds of the software that ran News Feed, the C++ binary responsible for ranking the posts in the news feed in particular. We were particularly interested in performance regressions, new crashes arising from new race conditions or configuration mismatches in the Thrift data structure definitions, and impact on advertising revenue.

My main project on this team was to add per-diff address sanitization (ASAN) canary tests, to surface these problems while the developer was working on their code, rather than waiting for the CI process to catch it. This involved working with several teams, including the Sandcastle team (CI), Phabricator team (code review + test maintenance), securing server capacity, and deploying it into production.

Independent of this, I was also on the push oncall rotation for Aggregator, which involved identifying regressions in the categories previously mentioned, and working with SWEs to identify and fix the root cause issue.

IMVU, Redwood City, CA— *Software Engineer I*

November 2016-September 2017

IMVU's FIRE team was primarily tasked with identifying and fixing problems in a diverse codebase, primarily PHP and Haskell, but also including CSS and JS. My projects involved extending and maintaining a customer marketing tool that would extract information about users, such as if they had spent money, when they joined, if they had a "special someone", and so forth, and offer targeted promotions based on that information. This also involved extending the UI the marketing team used to put together these marketing promotions to support the new predicates they requested.

I also designed, implemented, and tested a backup-and-restore system for Scylla, a C++ re-implementation of Cassandra, to replace a large vertical Redis shard acting as a data store for our News Feed endpoint. This involved a sizable amount of experimentation and reverse engineering, as the documentation for this was fairly slim.

The FIRE team also worked closely with Ops to help triage and mitigate site outages, as well as participate in post-mortems to identify root causes, and take follow up tasks to implement appropriate fixes.

TheTangoCoder@gmail.com
github.com/TheWizardTower
linkedin.com/in/adammccullough

Skills

Analytical and Resourceful
Functional Programming

Autodidact

Reflexively curious

Excellent communicator

Mentorship

Solution seeker

Passion for Programming and Engineering

Accomplished in trouble-shooting root causes to customer impact

Functional Programming

Linux Administration

Languages

Haskell, C++, Scala, Golang, Perl, Python, PHP, Bash.

Volunteer Work

Organized and Ran a Haskell Class at IMVU, 2016

President, ASULUG, August 2010-August 2012

Personal achievements

Amateur Extra Radio license, callsign AG7YC

IMVU, Redwood City, CA— *Systems Engineer II*

June 2013-November 2016

IMVU's ops team managed the operations side of the outfit, including managing CPU, memory, and storage capacity, ensuring correct and complete configuration management, data retention SLAs were met, and effective stewardship of our monitoring and paging solutions. While oncall, we were expected to identify and manage mitigation of site issues our monitoring alerted us to, and communicate the current situation, and steps we are taking to mitigate, company wide. We also worked with other teams in the company to identify issues affecting production health and assist them in improving their service quality.

Much of our work was interrupt driven from shepherding out production cluster, however my largest project based contribution was replacing the caching load balancer used for our server side image rendering from Varnish to Apache Traffic Server. This resolved a memory leak that Varnish exhibited when running on our prod version of Ubuntu. This resulted in both more reliability and a significant reduction in oncall load.

Our prod cluster included Memcache, Redis, MySQL, Apache running PHP, Nginx, HAProxy, as well as static Haskell binaries. We used CFAgent to provision and maintain production hardware configuration, and postmortemed service quality incidents to identify root causes, then identify and implement follow-up fixes.

SMART Storage Solutions, Chandler, AZ— *Software Engineering Intern*

August 2009-February 2013

Developed hard drive test software. Designed and deployed a test suite for the backplanes used to test said drives. Refactored a drive erase program and expand functionality for customers. Assisted in running and developing a flash characterization test/burn-in procedure using both hardware, embedded firmware, and software written in the .NET framework.

Education

Arizona State University, Tempe, AZ— *Computer Systems Engineering*

August 2007-May 2013

Speaking Engagements

Lambda Calculus for the Easily Confused

Speaker - LambdaConf 2017

Walkthrough of the mechanics of Lambda Calculus, with a particular emphasis on how these mechanics can give an intuition for things in Haskell, like partial function application or higher-kinded types. Concludes with a demonstration of the Y combinator.

Monad Transformers for the Easily Confused

Speaker - LambdaConf 2018

Begin with a review of the Monoid, Functor, Applicative, and Monad typeclasses. Demonstrate that Monads do not compose, then demonstrate how Monad Transformers solve this problem, first with the IdentityT monad, then the MaybeT monad.

Rust’s Borrow Checker Proven Correct

Speaker - LambdaConf 2019

Gave an overview of bugs in various compilers, and how difficult they are to identify and debug. Describe how the Rust devs used formal verification to prove that the semantics of the borrow-checker would produce programs without race conditions. Conclude with some motivating examples to demonstrate why various designs fail the typechecker, and what race conditions they prevent.