Dan Pittman

Programmer and technical product developer
503.980.5841 | dan@dpitt.me | 2CC7 8581 8E79 5244

Auxon Corporation :: Principal and Founding Engineer

07/2018 - Current

For the first half of my time at Auxon, I continued my exploration into applying proof theoretical methods for checking software at both compile time and runtime.

Once Auxon moved on from developing high-assurance software itself, I began focussing on the customer-facing edges of our product, finding that putting myself in the shoes of potential users came naturally. During this time I developed and implemented Modality's `log`, `check`, and cumulative report features.

Additional Accomplishments

- We made a runtime in Rust on top of seL4. In this runtime, we went through great lengths to extend the guarantees that seL4 provides to also be checkable at compile-time through techniques which can be summarized as "type-level programming".
- I owned the architecture-agnostic implementation for that runtime that used type-level induction to provide a mechanism for architecture-agnostic memory hierarchies.
- I Designed and implemented a memory-mapped registers library (in Rust) that provided:
  - Compile-type checked writes (for values known a priori).
  - A proof-carrying approach for runtime checking.

  - Complete proofs of that library's correctness in Agda.

- I built a tool in Haskell that translates Rust types (and therefore type-level programs) to Agda. I spoke about an early iteration of this at Strange Loop 2018.

Polysync :: Principal Engineer

11/2017 - 07/2018

I came to PolySync to contibute to the high-assurance automotive runtime they were building working as a hybrid proof / software engineer. PolySync intended to use my work and research to find ways to blur the line bewtween proof and program to instill further confidence in the integrity of the software they developed.

Additional Accomplishments

- We made a toolchain for writing seL4 tasks in Rust.

- I designed a distributed system whose core logic was session types predicated on Atkey-style indexed monads translated to Rust to run on seL4.

Further Experience

Intel :: Sr. Cloud Software Engineer

12/2014 - 11/2017

- I led a team that built a distributed execution engine that tried to model the data center as a category—states as objects, transitions as morphisms (ask me more about this, it was fun).

- I worked on making persistent memory useful in cloud environments (Go/etcd).

- I helped design and build a modular and flexible telemetry system for use in the data center.

VMware :: Sr. Member of Technical Staff

12/2012 - 12/2014

- I was one of the first 10 developers working on VMware's public cloud platform, vCHS.
- We made a globally distributed system which could deploy complete private VMware environments to the cloud in minutes. It did this through a number of interesting technologies including but not limited to: CQRS & Event Sourcing, AMQP, Cassandra, (and all of its finicky eventual consistency idiosyncracies), and a homegrown Paxos implementation—which I was later responsible for replacing with Zookeeper.