290 days at Adtran

John Hossler

December 1, 2017

As of today, not taking into account sick time or vacation time I have taken because I won't put THAT much work into a title, I have spent 290 days at Adtran. In this report, I am going over what I learned and what I accomplished during my 290 days. I have been co-oping at Adtran since the Summer of 2016 and will be finishing up my co-op at the end of this Fall 2017 term.

During my first term in ENPQ my Mentor was Salim, my Manager was Susan, and I shared an office with Trevor Murphy.

Being on this team gave me the opportunity to work with several technologies, such as python, ruby, docker, robot framework, and vWLAN.

While on this team, I worked on the TAUT Libraries. TAUT is a set of packages meant to centralize tools used to test products, because before everyone would develop their own way for interacting with devices. It lead to a lot of repetitious code because of low visibility into other options. At the time, this package was massive. Not only did it have python files thousands of lines long, it also had a large numer of files and a complex directory structure. One of our tasks was to rework these libraries to follow dry principles. A lot of these packages

defined the same behavior in multiple places, and by applying DRY principles, we were able to cut down on that repetition. As you can see here, we separated the libraries out into platform definitions, command libraries, and communication libraries. In addition to some rework in structure, we also reworked the libraries to follow PEP8 standards. This task ended up taking most my time because it went beyond what normal autopep8 packages would do: class names and variable names needed their case fixed, as well as dictionary kes. There was also an issue with dictionary keys needing to be updated to follow naming standards. I was able to fix a lot of these issues by writing a script to do it, but because of low unit test coverage, it was difficult to be confident in these changes. We also added new platform definitions.

I learned a lot my first term. I went from knowing basically nothing about Adtran to gaining some basic understanding of the products Adtran offered. I started learning about coding standards and why they are beneficial to codebases. This was also the first time I used Git in a professional setting (not just with friends in a class), so the management of branches and merging was mostly new to me. I also had never gone through any code review process before, so I learned how to use code collaborator and became less scared to have other developers critique my code.

On my second term at Adtran, I was put on MVT. Don't ask me what MVT means right now, because I don't know, but I do know the meaning has changed over time. While I was on MVT, I had TWO mentors: Brandon Leatherwood for much of the spring term, and then Casey Small after that until he left Adtran. By that time, though, I was integrated into the team enough that a mentor wasn't necessary. My manager was Matt Moody, and I was lucky enough to spend the spring and summer with this team.

I worked with a lot of tools on this team that I had worked with before on my first term, like python and robot, but I also got the opportunity to work with packaging in python, python scaffold, netconf, and jenkins.

While I was on this team, MVT owned the MOSAIC OS CI Product Test Pipeline. The goal of this was to automate acceptance level tests to ensure that new checkins didn't break components. Frequently, they did though, and part of our jobs was helping the developers debug the issue so the pipeline would be green again. We also added functionality so more test cases could be ran on our test beds, we helped test writers convert their robot tests so that they could utilize our tools and have their tests automated, and we built testbeds for these tests to be ran on. Ultimately, our goal was to ensure that the pipeline stayed green as much as possible, and add more tests so that the meaning of a green pipeline meant a more stable product.

During my time with MVT, I had a lot of opportunities to learn about tools and styles of programming. I learned about python virtual environments, python packaging (including deploying to pypi, automated tests), how to use the Adtran Python Scaffold, unit testing principles (like what to test, how and what to mock, etc), and test driven development. At the beginning of my term, I asked my mentor, Brandon Leatherwood, if I could try to practice Test Driven Development on my task. After getting confirmation, I worked at it, I read about how to do it, and eventually I settled into a rythym. Since then, I've been using Test Driven Development any time I work on a package, and I think by being given that opportunity to practice Test Driven Development, I was made a better developer.

My last term at adtran has been spent working on TARP, a DEVOPS team. My mentor is Robby Pocase, and my manager Rick King. This team was the first team I worked on

where I worked in a collaborative team space, and I had a positive experience of always being so close to my team. I would definitely recommend being on at least one team in an open team space during co-op. Another interesting thing about this team is that I didn't do as much software development as I had on other teams. Instead, most of what I was doing was working with managing services, and while sometimes code was necessary, other times the job was more about doing something that a service allowed you to do in the first place. For the most part, I was helping manage Artifactory, Github, and Jenkins, but I also got to work with both Python and Groovy.

With Artifactory, I learned how to create repositories, deploy artifactory instances, and I came up with a promotion model for a group to fit their needs. The promotion model ended up teaching me a lot about Artifactory and respositories; with the model I provided, there was a way to query for the latest stable build and promote from scratch to stable easily. A lot of the work was demonstrating how the promotion could be done. For this, I worked with jenkins pipeline scripts and interacted with the artifactory plugin directly to publish or promote content. I also needed to interact with the REST API to query for the latest version of an artifact.

I learned a lot about Github administartion as well. In one task, I needed to create a custom bot authentication token in order to inject those credentials into a bot to automate some of the github administration, like creating repositories, organizations, and teams. A lot of this automation centered around a python package called github-admin-py, which interacts with the github API to perform actions for a user. I added the ability to assign teams to repositories, as well as the ability to get a list of collaborators for a repository. This helped automate some tasks, like migrating adpackage jobs to jenkins enterprise and

syncing permissions between github repositories and corresponding jenkins jobs.

That brings me to the last tool that I used heavily on TARP. Jenkins, as some of us may know, is a tool that allows for some task to be automated, like running tests, or building binaries. One of the first issues with jenkins I faced was setting up periodic reboots of the computers that run the jobs without disrupting other jobs. We had security updates happening frequently, but some updates needed a reboot to take effect. To accomplish this task, I used jenkins to have a script run weekly that would take each node down, and make sure it came back up. Another issue that we had was that jobs that were generated by scanning github did not have their permissions automatically synced, meaning the people who had permissions to modify the repository on github would not have permissions to run the job on jenkins. I worked with members of my team to help set up some synchronization on a weekly basis, and then recently created another self service job that users can run against a specific repository. I also was able to take advantage of my earlier additions to github-admin-py to automatically assign the correct teams to newly generated repositories.

Now that my time as a co-op at Adtran is over, it's time to look to the future. In this upcoming spring, I will be graduating from the university of alabama with a major in Computer Science, and that summer I will be coming back to adtran to work full time for MVT. I'm excited to come back to Adtran, and while I'll be in a new position, my co-op experience has given me the confidence to know that I will be able to contribute to the team and I know I'll enjoy what I'm doing.