MECOP Internship Report

ThermoFisher
SCIENTIFIC

Alex Grejuc

Mentor: Kevin D'Souza
kevin.dsouza@thermofisher.com
503 726 2705

Computer Science
First Internship
September 5th, 2019

Contents

1 About Thermo Fisher Scientific

Thermo Fisher Scientific is a global scientific service and product development company of 70,000 employees and revenues exceeding $24 billion. It is made up of five brands: Thermo Scientific, Applied Biosystems, Invitrogen, Fisher Scientific, and Unity Lab Services. Together these five brands address "a range of needs from sample and material characterization" [1] (e.g. electron microscopy), genetic analysis, molecular biology technology, lab equipment, and lab services.

In 2016 Thermo Fisher Scientific acquired FEI (Field Electron and Ion) Company, a business which specialized in electron microscopy. The Hillsboro office in which I worked was the former headquarters of FEI. It is still the location for much of the research and development at Thermo Fisher Scientific. Customers of this site include companies from the top tier of the semiconductor industry.

2 My Role at Thermo Fisher Scientific

I worked as a software development intern on various projects related to electron microscopy in the New Product Development (NPD) section of the Research and Development department of the Hillsboro office. As part of this role I developed new features, fixed bugs, refactored code, wrote tests, gathered requirements, prepared techinal reports (for proposed software changes), documented features, and delivered presentations.

My internship coordination was managed by Kean Stump, a Senior Manager of Software Engineering. Throughout my internship I was mentored by Kevin D'Souza, a Hillsboro NPD Platform Software Manager. This mentorship included weekly one-on-one meetings in which I could ask questions, update on my progress, learn about the company, and get acquainted with the office.

For the initial four months of my internship I worked under Konstantin Balashov, a Software Engineer in the NPD FIB Software Center of Excellence team. The bulk of my work with Konstantin was for FORT (Framework for On-tool Regression Testing), a command language used within the company for alignment testing, automation, and microscope control. I also contributed to the Tool Readiness (microscope alignment service) software and did some DevOps work to assist in debugging crash dumps from the field.

In the latter portion of my internship I worked under Michael Moriarty, a Software Quality Engineer in the NPD Platform Software team. During this period I worked on a command line utility to parse, analyze, store, and retrieve software test output.

[1] Quote retrieved from https://www.thermofisher.com/us/en/home/brands/thermo-scientific.html

3 List of Projects

- FORT feature development and bug fixes
- FORT Transport network application
- Debugging symbol management
- Tool Readiness image analysis graphical application
- Test Analysis command line utility

4 Executive Summary

4.1 Major Project Overview

4.1.1 FORT feature development

I worked as a FORT feature developer for all four months with Konstantin during which time the other projects were intermittently woven in. As described previously, FORT stands for Framework for On-tool Regression Testing, and it is best described as a command language for running alignments and executing other hardware and software actions (such as stage moves and image grabs) on the microscope.

4.1.2 Fort Transport

I implemented a transport mechanism made up of three WCF components: a transporter (for receiving files), a scheduler (for running jobs), and an agent (a user interface where files are provided and jobs are requested for running). Its purpose was to support automation by remotely running FORT jobs and also to support testing of the Tool Readiness alignment service by allowing users to provide different binary files of the service against which to execute their FORT jobs.

4.1.3 Image Analysis

I extended a Tool Readiness image analysis utility in several phases. One set of changes was the user interface changes, such as search functionality, keyboard shortcuts, and the option to run bulk operations for automation purposes. Another set of changes was to more thoroughly display the results of image analysis to the user and add support for exporting this to CSV and image metadata files. Finally, I also drew the results of the analysis on images and overlaid the results of various analyses over each other.

4.1.4 Debugging symbol management

I changed the software build process to ensure that PDB files were saved to a server for every version of 15 different software repositories. Along the way I helped fix a problem regarding overwritten files and documented the process for other developers.

4.1.5 Test analysis utility

During the remaining six weeks of my internship I worked on a test analysis utility to improve the software quality assurance efforts at Thermo Fisher Scientific. For this project, I worked on a command line application for parsing different types of test output files, storing them in a database, and interfacing with that database.

4.2 Results of Work

The result of my work on FORT is that system engineers, software quality specialists, and other employees within the company have an improved workflow for running, developing, and testing alignments. My work on the image analysis utility achieves a similar result in assisting scientists and engineers in understanding and developing crucial metrology methods which assist in microscope alignment.

My DevOps work led to an updated build process for some of the repositories at the company. This will support troubleshooting in the future. By having all the necessary debugging files, software engineers at the company will be able to debug crash dumps from the field in-house.

My work on the test analysis utility will assist software quality engineers in tracking test history, identifying root causes of test failures, assigning issues to engineers, and ultimately in fixing bugs.

Altogether, my six months of work, albeit in a small way, helped advance progress in the field of microscopy and in turn helped advance science.

4.3 Accomplishments

I implemented nine new FORT features, worked on five different projects, prepared a proposal document for a software change which was approved, received positive feedback in a code review, delivered two presentations to the software group in the Hillsboro office, and created two new software projects from the ground up.

5 Projects

5.1 Fort Development

FORT is a simple interpreted microscope command language which has features for hardware control of the microscope (such as stage movements or milling, primarily for the purpose of executing, testing, and developing microscope alignments) as well as software control (such as file I/O, image grabbing, and database storage). It allows engineers and scientists at Thermo Fisher to focus on applying their domain knowledge rather than programming since the details of a fully-featured programming language are abstracted away.

My work as a FORT developer gave me a full slice of what I will do in my career as a software developer. I implemented new features in an existing code base; attended meetings to plan, discuss, and provide updates on my work in an agile development setting; wrote some (but not so much as to extend beyond my duties as a developer) documentation for what I implemented; fixed bugs, and refactored code.

I implemented nine new features including one for moving the stage, one for grabbing and saving images, one for milling a specific pattern used to diagnose alignments, and updating an multi-application interface to increase the efficiency of movements. I also fixed bugs related to parsing, alignment termination, and microscope disruption reaction. Alongside these changes I refactored code as needed to prevent code duplication, improve logging, and even to switch out and wrap up a core data structure with a new class.

Throughout this process I documented all of my additions in a language specification document and wrote simple example scripts to demonstrate their use. Finally, on occasion I accompanied Systems engineers to help troubleshoot problems related to FORT or to see the results of my development in action.

5.2 Test Analysis Utility

In the final six weeks of my internship I developed an application for parsing test output files, analyzing them, storing them to a database, and providing a user interface for accessing the database. Part of this effort was a hash which can be configured to combine desired parts of the test output to group different instances of test failures with the same root cause. The purpose of this utility is to improve the workflow of software quality engineers in their process of understanding and tracking test failures, identifying root causes of bugs, assigning them to developers, and streamlining the software merge process.

Given that my work surrounded test infrastructure and that Michael is a Software Quality Engineer, writing tests for this project was emphasized. After some initial requirements gathering and prototypical programming I began doing my work in a test-driven development style which I found to be instructive and incredibly useful (especially when extending or refactoring my code). Additionally, I picked up some basics of SQL database design and usage as I had very little prior knowledge on the subject.

6 Conclusion

Throughout the course of my internship I matured as a software developer, as a software engineer, and as a professional. As a software developer I added C#, Windows development, test-driven development, MySQL, WCF, WPF & WinForms, as well as batch & PowerShell scripting to my toolkit. Of these skills I made the most significant progress as a C# developer: prior to the start of the internship I had never written a line of C# code and now by the end of it I am confident and comfortable using it to develop useful software. As a software engineer I have improved my ability to solve problems and to do so with extensible and understandable designs. I now have real-world experience in designing software from the ground up (as I did on the FORT Transport and Test Analysis projects) or extending old code and in both cases I am able to spot faults in my thinking sooner than ever before. Finally, as a professional I improved my abilities to communicate, prioritize, and collaborate. I got better at presenting technical information; clarifying my understanding as soon as needed; knowing when, whom, and how to ask for help; participating in meetings; and holding myself to a high standard. From all of this Thermo Fisher Scientific benefited by having a motivated and diligent employee whose effort helped to streamline the work of employees within the company in their mission to innovate, discover, and develop scientific methods and technologies.

7 Jargon Definitions

- FORT: Framework for on-tool regression testing; a command language used within the company for alignment testing, automation, and microscope control.
- PDB: Program database, a Microsoft file format for storing debug symbols (such as identifier names and line numbers) which provide developers with helpful information for troubleshooting software.
- Regression Test: A software test intended to verify that a code change has not caused a regression in functionality (i.e. breaking features or bugs).
- .NET: A Microsoft software framework which provides language-interoperable libraries and execution on a virtual machine.
- WCF: Windows Communication Foundation, a .NET framework for creating applications which communicate as web services.
- WPF: Windows Presentation Foundation, a Microsoft user interface framework for desktop applications.
- WinForms: A Microsoft user interface framework for desktop applications.
- Crash Dump: A snapshot of the state of a computer at a given time which includes the contents of the computer's memory and registers as well as operating system information.
- DevOps: Work that is at the intersection of IT and software development and includes the management of areas such as software building, deployment, and integration.
- Parse: To break up, simplify, or understand one type of structured data into another. Within computer science this is for the purpose of improving the ease of data manipulation and is implemented by a program called a parser.
- Repository: A storage location for a software project, almost always backed by an implied version control system which enables developers to easily save, merge, or revert changes to a code base.
- Stage: The portion of the microscope on which a sample is placed and observed from.
- Test-Driven Development: A software development methodology in which tests are written before the code. In practice this is done in a feedback loop of writing the minimal portion of test which makes the method or component of code fail, then writing the code to pass the test and looping until the desired result is achieved.
- FIB: Focused ion beam, a microscope similar to a scanning electron microscope but which uses ions in place of electrons.