Thomas L. Byrne

REGENTS BACHELOR OF ARTS PROGRAM

PORTFOLIO TO BE EVALUATED FOR COLLEGE EQUIVALENT CREDIT

SPRING, 2014

Table Of Contents

Master Course List

Marshall University Credits Requested:

<u>College Of Information Technology & Engineering</u>

CS 305 - Software Engineering 1
CS 320 - Internetworking

Tom Byrne

51 Southgate Court
San Jose, CA 95138

T 408.391.9307
kylar42@gmail.com

PROFILE

Expert in Java, large scale web systems and HTTP.

SKILLS

Expert in: Java, Swing, HTTP, WebDAV, Multithreading, Object Oriented Design, API Design

Also Skilled in: XML, JAX, UML, SDLC, C, Python, Ant, Technical writing

NOTABLE ACCOMPLISHMENTS

Contributing author and technical editor for 'Configuring Netscreen Firewalls' - ISBN 1932266399

Speaker, Enterprise Service Bus Integration, The Server Side Symposium, 2010

Co-Author, BookmarkDAV specification for common Bookmark Exchange (Pending submission to IETF)

EXPERIENCE

Software Architect, IDS/FaceTime iCloud, Apple Inc; Cupertino, CA 2013-Present
Architect working on scalability of internal platforms for multiple services.

Research & Design for advanced & new projects.

Software Architect, iCloud, Apple Inc; Cupertino, CA 2010-2013
Current PIM (Personal Information Management) architect for all of iCloud - http://www.apple.com/icloud/

Evaluated multiple technologies and frameworks for large scale server systems including Nonblocking IO, Multithreaded servers and NoSQL.

Wrote protocol servers for synchronization of personal contacts, bookmarks.

Co-Authored specification for Bookmark specific DAV extensions using XBEL & XML

Implemented multiple IETF specifications, including RFC6352, RFC4918, VCard, WebDAV Collection Sync for large scale web service

Used HTTP, WebDAV, Multithreading, Memcache, Google Protobuf, Java, Python

Senior Software Engineer, MobileMe Group, Apple Computers; Cupertino, CA 2006-2010
Worked on second generation of HTTP/WebDAV server platform (iDisk), Lead multiple features to completion, did API & ASI design for multiple teams, as well as integration with multiple groups.

Designed & implemented photo gallery server publishing API's and ASI's for MobileMe Gallery.

Responsible for high level design & architecture, down to full implementation of multiple features.

Used Java, C, HTTP, WebDAV, REST, BerkeleyDB, Solaris

UI Architect, Packeteer; Cupertino, CA 2005-2006
Technical Architect on User Experience team - worked inside a team to define features & user interactions for a next-generation Traffic Management product.

Designed, architected and implemented API's, frameworks and complete UI prototype.

Worked with server & database teams to develop consistent API's for multiple clients.

Lead a team of 3 engineers to on-time delivery of Alpha and Beta of product.

Used Java, Swing, Postgres, JAXB, Ant, Fedora Core Linux

Senior Software Engineer, Netscreen/Juniper Networks; Sunnyvale, CA 2001-2005
Technical lead on the team that produced the first and second generation of Netscreen's Security Manager software, from design through shipping: http://www.juniper.net/us/en/products-services/software/network-management-software/nsm/

Specifically worked on security features, architectural & GUI frameworks, packet level parsing and High Availability features.

Used C, Java, TCP, UDP, HTTP, RMI, UML

Senior Software Engineer/UI Lead, ePatterns.com; Palo Alto, CA 2000-2001
Defined & Implemented portions of a Business Process Management & Analysis system.

Lead a small UI group through 2 prototypes and alpha revision of a product, used JDK, Swing, EJB's and Orion J2EE platform.

On cross-functional team to define inter-service API's

Software Engineer, Lightsocket Inc; San Jose, CA 2000
Worked with a team to develop real-time collaboration software to run on a hosted server platform, used Java 1.2, 1.3, Swing/JFC

Also acted as company's ad-hock system administrator for 5 months, duties included hard wiring ethernet, switch & router configuration, Desktop & PC setup & troubleshooting, Windows, Linux & Solaris administration, including DNS & BIND, Sendmail, Apache, Windows 2K Domain administration.

Software Engineer, Petrasoft Research; Santa Clara, CA 1999-2000
Worked with other software engineers to design a Point Of Sale system, using Java 1.2, JFC/Swing, JDBC, Sybase 11

Software Contractor, Essex County Board Of Education; Essex, Ontario 1995-1998
Designed & Implemented a set of educational software for grades 4 through 8 using Microsoft C++, Microsoft Foundation Classes and Borland's C Builder tool suite.

EDUCATION
Regents Bachelor Of Arts, West Virginia University at Parkersburg 2013-Present (In Progress)

Associate in Applied Science, West Virginia University at Parkersburg, Awarded 2013.

Computer Systems Technology, Mohawk College, Hamilton Ontario 1993-1995

Educational Narrative

I had always planned to go to university. I knew from a fairly early age that I wanted to pursue a career in a scientific field, and considered varying hard sciences like math (which I loved), chemistry, and physics among others. My grades did not reflect my abilities, and after scoring as genius level on a school IQ test, I was put into an advanced learning program for the remainder of grade school. Despite this, I had a steady stream of C's and B's. I could attribute this to many things, but after evaluating it over time, I think that it primarily stemmed from two things: boredom, and a lack of organizational skills. I didn't realize until I was in my late 20's that I had a moderate attention deficit disorder that went undiagnosed (My two younger brothers would later be diagnosed with severe ADHD), and that hampered my abilities to succeed in school. Regardless, I doubt I would have been successful in a standard, structured school environment at the time, as I was quite ambivalent about most of the material being taught. When it came to any topic that I was interested in, I took a self-learning approach and would spend hours or days devouring any books, articles, or other media I could get, to the point where I would be far beyond my grade level when it came to that topic. This pattern continued until high school, where I flunked out of my first high school, then spent a semester at an alternative method high school (where I earned the highest grade the school had ever given out in math) and finally graduated from General Amherst High School in January of 1994.

In Ontario, until 2004, students wanting to pursue a 4 year baccalaureate degree were expected to graduate with 30 credits, including 6 Ontario Academic Credits(OAC's) , which are generally taken during a fifth year of high school. This was previously just referred to as Grade 13 until 1984. When I graduated, I had only 2 OAC credits, thus deterring my plans for a 4 year post-secondary education. While looking at my options, I considered several 2 and 3 year Art & Technical colleges (equivalent to a junior college or community college in the US) and applied to several. I was accepted to Mohawk College of Applied Arts & Technology in Hamilton, Ontario, about 3 hours from where I was living with my parents. In the fall of 1994, I moved into a small apartment 2 blocks from campus, and began my college education.

The transition from living at home to living on my own and being entirely responsible for myself made for a tough transition. I had never had good study habits, and had instead mostly 'skated by' in high school. I was entirely unprepared for success in college, and it showed. I spent a lot of time watching TV, skipping class and playing varying Role Playing Games. I still managed to show up for classes and exams, but I wasn't engaged in my education. When my second semester rolled around, I wasn't even able to keep up the facade of being interested in my classes. This resulted in academic probation, and eventually I was dropped from the program when I failed 4 of 6 courses.

I moved back home and worked part time for the next few months, trying to get a job in the field where I was most interested (computers), but was unsuccessful. The area where I was living (southern Ontario, near Windsor) was a primarily industrial area, and any technology jobs were fought for by multiple, often overqualified applicants. Despite my self-taught computer abilities, I always lost out to older applicants with a degree. By the fall of 1995, I was determined to reapply and finish my degree. I was re-accepted into my degree program at Mohawk and moved back in January of 1996. I successfully completed the following 2 semesters, and had a co-op placement at the Royal Bank of Canada in Toronto as a paid intern in their internet banking division. This was the first time where my ability to self-learn came into play in a job. I was thrown into several technical tasks which I had no background in, and was forced to learn very quickly to succeed there.

In 1997 I returned to Mohawk for the last semester of my 2nd year, and fell back into old habits, and became uninterested in school. I flunked all my courses and left school. I spent 1998 applying for the very few technical jobs I could find, and being rejected, while working part time as a music teacher (guitar, beginning piano and introduction to music theory), and also as a security guard, and being generally unhappy. In late 1998, I was speaking to a friend of mine who lived in California, and was generally complaining about my life and how there was a lack of jobs that I was interested in, when she made a statement that would ultimately change my life: “Why don’t you move here, to California? There are 5 jobs for every person, and most companies can’t hire fast enough.” Of course, the first thing that I thought was “That’s stupid. I’m not going to pack my meager belongings into my car and drive 2600 miles to a strange place and try to find a job.” The more I considered my other options, the more appealing it seemed. In January of 1999 I packed a few things and a box of food into my car and drove to California.

I spent 3 months living in my car before finding an apartment with two other people from church. I spent 5 months doing odd jobs wherever I could find them and working as a temp at a local day program for disabled adults to pay the bills, and attended job fairs, hiring fairs and going on interviews. Finally in June of 1999, an acquaintance from church hired me as a junior engineer at his company Petrasoft Research, which was soon acquired by a larger startup company Lightsocket, Inc. While employed there, I attended an Introduction To Java Swing course at Evergreen Community College to introduce me to Java and it’s GUI tools. I also attended several JavaOne conferences and was able to learn about many technical tools and techniques, including instrumentation and industry best practices. I still was more of a 'hacker' in my knowledge - using the quickest way to solve problems, instead of thinking through them.

In August of 2000, I married my wife Kelly, and upon returning from our honeymoon, found that Lightsocket, Inc had run out of money and was laying off half it's employees. I was lucky to remain, but the company ended up going out of business in October of 2000. I was now stuck with a new, expensive apartment lease and looking for a job. I was very fortunate to find a great position at a very small, new company called ePatterns, from the second job that I applied for. This position was another where I was thrown in with insufficient knowledge and was forced to learn myself very quickly just to stay afloat. The company was founded and staffed largely by graduate students(and two professors) from Stanford's computer science division, and nearly every discussion that I was involved in ended up with me reading up on software and computer design concepts and implementations. During my time at ePatterns I learned more about the academic side of computer science and computational theory than in the entire time I attended college. The company didn't succeed, and in fact went under a mere 7 months after I joined due to a lack of venture funding. My wife and I had just found out that she was pregnant, and I was again looking for work.

This time, the hunt for work was different. I had learned and accomplished so much in the previous 7 months that I was now well prepared for a job in the industry. Instead of searching online job ads and the newspaper, I called two people I had previously worked with and asked if there were open positions at their companies. This resulted in two immediate interviews and two job offers within 10 days. For a variety of reasons I chose the offer from Netscreen Technologies. I worked at Netscreen for 4 years, from May 2001 until May of 2005. While there I attended another JavaOne conference, concentrating on learning about new trends in User Interface programming, and the latest version of the Java programming language. I then was able to transition our entire team to this new version with virtually no interruption or difficulties.

Netscreen was acquired by Juniper Networks in 2004, the month before my son Christian was born. I made the transition after the acquisition, but found that I wasn’t enjoying my work as much as I had previously. We went through several mid and senior management changes and I began to consider leaving. In the spring of 2005 I was approached by a company who had found my resume on my website. I kept my resume up to date and available for anyone to view as an ongoing exercise, and it certainly paid off in this case. Packeteer was forming a new group to build a brand new project from the ground up, and was planning on utilizing several new processes that I had recently been reading about, including Agile/Scrum and User Experience(UX) & Interaction Design. I interviewed there, and met with members of the team, and found myself very excited to join their team, which I did in May of 2005.

This position again required me to learn a great deal about the new UX processes that my immediate team were using, and find ways to translate those into the technical processes that the larger technical team were using. Our team was put through a week long course on Interaction Design training by Cooper Interaction Design. Interacting with the server engineers also taught me a great deal, as they were very heavily invested in using computerized tools for automating their development processes. I attended an industry convention in March 2006, The Server Side Java Symposium, where I focused on learning about using tools to streamline a software development process, as well as communications protocols for Java. In the spring of 2006 our company went through some senior management changes which trickled down to our group and we were told that our project was being cancelled. In another twist of fate, I was approached that same week by a former colleague at Apple, Inc, who had an opening in his group: .Mac.

Apple was, and had always been a place where I wanted to work. They had long been groundbreaking in the industry, and were often considered to be the 'shining star' of one's career. I had previously applied twice, once in 2000 and once in 2001, and had not even managed to get an interview, so needless to say, I was thrilled with the opportunity. The position, however, was one that I was only mildly qualified for, it was entirely server work, whilst I had spent the last 6 years doing primarily graphical interface work. I was familiar with the base technologies, but once again, this job would require a lot of learning. I joined Apple in May of 2006 and have been there ever since, moving from Software Engineer to Senior Software Engineer, and finally to Software Architect, where I was one of the first people to work on iCloud and followed it through it's entire life cycle. At Apple I've had to learn everything from operating systems to databases to project management. I also attended the Java Server Side Symposium in 2010 and 2011, focusing on cloud technologies and large scale deployments of web servers.

I have regretted not getting my degree for several years, but have not had the time or opportunity to go back to school until I moved to Ohio in October of 2011, and enrolled at WVUP Board Of Governors program, with the intent of continuing on to the Regents Bachelor Of Arts program and getting my degree. I want to finish my degree for two main reasons: first, to finish something that I started and prove to myself that I'm capable of it. Secondly, I have spent a great deal of spare time working on a project that I enjoy that involves using computer science to solve difficult math problems, and I would love to take higher level advanced math classes to help me with that project. I finished my Associates of Science degree, and am very close to finishing my Bachelors, specializing in Computer Science. I am nearly finished (I need 6 credit hours to graduate, as of this writing), and am already looking into post-graduate work.

Work Narrative

Music Instructor, Musicland
Location: Amherstburg, Ontario, Canada
Dates: 1992-1996
Reported To: Mrs Farmer, Store Manager
Skills: Introductory Music Theory, Piano Instruction, Guitar Instruction

I worked part-time at Musicland during summers and after school from 1992-1996.

During this time, I was a guitar instructor for all levels from beginner to advanced. I scheduled students on a weekly and per-semester basis, based on our mutual availability and convenience. I taught guitar fingering, beginning theory, chords, transposition, ear training, sight reading, improvisation, scales, and rhythm structures.

I taught introductory piano and music theory as well; teaching triads, scales, modes, harmonizing, transposition, sight reading, basic rhythm and basic melodies.

Software Contractor, Essex County Board Of Education
Location: Essex, Ontario, Canada
Dates: 1995-1998
Reported To: John McLaughlin, Teacher on special assignment
Skills: Authority Software Package, C, Microsoft Foundation Classes, Borland C Builder, Photoshop, Windows 95

From 1995 until 1998, I worked part-time as a contractor for the Essex County School Board. The school board had an initiative to create computer-based activities that corresponded to the province's common curriculum. I was contracted to create the first versions of these activities.

The school board had licensed a software authoring package named 'Authority'. Using this package, I created several dozen Windows applications for grades 4 through 8. These applications covered various math concepts like multiplication and division. I also created applications for a variety of language arts programs that covered core concepts like sentence structure, word definitions and identifying nouns, verbs, adjectives and adverbs.

The next year, as my boss and I continued the program, we found that the Authority package did not allow us the flexibility we needed. We evaluated other development packages and chose Borland C Builder. Using C Builder from 1996-1998, we created more than a hundred additional independent activities and applications that are still in use today.

While creating these software applications, my boss would analyze the materials and give me sample activities from textbooks, workbooks or other sources. I would then design the application and Graphical User Interface (GUI) using Borland C Builder and Microsoft Foundation Classes (MFC) to be as close a match to the worksheet or source material as possible. This could be as simple as having the student type in an answer to a simple math problem, or as complicated as having them drag various shapes into a pattern.

Once we had developed a set of several dozen ‘activities’, we were able to rapidly create new ones by changing the data in them - for example, a simple worksheet of 10 multiplication problems could be easily made into 10 different worksheets, each with its own problems, by simply changing the data that was fed into the worksheet. This allowed us to cover a vast amount of curriculum with less work.

In January of 1999 I left this job, as I relocated to San Jose, California seeking full time work as a programmer.

Substitute Program Worker, Hope Rehabilitation
Location: San Jose, California
Dates: February 1999-May 1999
Skills: American Sign Language, Program Worksheets, Daily planning, Supervision.

I moved to San Jose, California in January 1999. While looking for a programming job, I took whatever part-time work I could find. I applied for a part-time temporary position at Hope Rehabilitation Services. I worked at Hope part time for 4 months, from February to May 1999. Hope Rehabilitation is a full time (5 days a week, 6 hours a day), community based day program for developmentally disabled adults.

As a temporary program worker, I would follow the instructions of the full time program staff. I assisted the clients in their day to day program activities. I followed clients' Individual Program Plans(IPP's). Each client had their own set of goals and activities, which I would implement. During outings, I would assist them in purchasing their goods, and supervise them while teaching them fundamental living skills. I would assist the clients in preparing their lunch, which included ordering food and cooking during special events. I assisted severely disabled clients with their personal hygiene and toileting, when necessary.

I filled out daily reports for any client that I was responsible for, noting their attitude, activities, progress, lunch and personal hygiene. Several clients and one program worker were deaf. Working side by side with them, I picked up a basic knowledge of American Sign Language(ASL). The program had a licensed instructor from the local college come once a week to teach ASL. I attended these classes for two months.

I left this position when I obtained a full time position as a programmer at Petrasoft Research.

Software Engineer, Petrasoft Research
Location: Santa Clara, California
Dates: June 1999 - December 1999
Reported To: Gregory Stone, Owner.
Skills: Java, Swing, Ethernet & Telephony wiring, Windows 95 & NT Administration, Photoshop, Database administration (Sybase, Microsoft SQL Server), Microsoft IIS Administration, JDBC

I worked at Petrasoft Research from June 1999 until December 1999 as a Software Engineer. My boss, Gregory Stone, formed Petrasoft, a contract engineering firm in June of 1999 and I was the second employee.

When Petrasoft was formed, we moved into a small building that was unprepared for a tech company. I spent the first month preparing the office for our company to move in. I ran category 5 cable (CAT-5) to each of four offices, a server closet, and to the basement where the telephone and data lines came in from the phone company. I had to wire each office with a data jack that ran to the server closet, as well as a phone line for each office. I hooked up all the wire in the server closet to an appropriate hub, and verified all the wiring and connectivity for data and telephone.

I was also responsible for setting up and administering a Windows NT 4.0 server for the office. I had to learn how to create a domain, add and delete users, and adjust user security privileges. I added and administered Microsoft Internet Information Services (IIS), and created and administered a database running Microsoft SQL Server for the 5 users that our office grew to.

I took a one week class in July: “Introduction to Java Swing”. I had a little experience with Java, but not its Graphical User Interface (GUI) toolkit, called Swing. This class provided me with a broad introduction to it.

Petrasoft took on a contract to re-write a retail ordering system for a local store. I was given the primary responsibility to write most of the application. This involved analyzing the old system, a 15 year old AS/400 with almost no documentation.

The first task was to get all the old data out into usable formats. With the help of our database programmer, we were able to dump all the old data to text files. From there, I had to design the new database tables and write code to normalize and clean the data. I was responsible for setting up a new server and database machine, installing Sybase AS 11, creating all the new tables, triggers and views for the new application. I then had to import all the data into the new database.

I then had to analyze all the workflows that the current AS/400 application was using. These were broken down into major groups of functionality: purchase orders, inventory management, data management, budgeting, sales management, reporting and administration.

Purchase orders required functionality to create, edit, delete and print purchase orders. Purchase orders are created to send to a vendor, but may be composed of products from multiple sources. Matching the data in the database to the correct vendors, with the correct unit prices, as well as matching the correct SKU numbers depending on the vendor and unit, required me to learn all about the procedures and processes for inventory management and ordering. I also had to write a printing framework so that the purchase orders could be printed out in the correct fashion for faxing to vendors.

Inventory management required functionality to track units that were present in the store or on order, to receive inventory when delivered (with or without a purchase order), or to manage the counts of inventory during periodic counting.

Data management required me to write UI to allow clerks to manage items in the database (such as adding a new item, or removing a discontinued one), manage prices on items (including sale prices, and the prices that were paid for the item from different vendors), as well as customer information.

Sales management required UI for listing the daily sales by department, and by register, so that the totals could be correlated with each cash register. At the end of every day, the sales from each register would have to be pulled to the central server, parsed and the data in the central database would have to be updated. This required a lot of database work. The inventories for each sold item would have to be updated, sales items would be created or incremented, and daily sales reports created.

Budgeting was a special request that was created for the new system, and didn't exist in the old system. The manager of the store wanted the ability to set a limit on the amount of purchasing that a department could do every quarter. This required me to write code that would look at all sales for that department for the last year, and calculated a weighted average for the current quarter. I also created a rolling calculation every month when the monthly sales reports were calculated. The manager could then raise or lower the budget for that department at will. The department managers were then able to create purchase orders up to the limit of their department budget without managerial approval.

Reporting was required by the general manager of the store in order to see all aspects of the store operations quickly. He needed reports to view the amount of inventory, inventory on order, month to date sales, year to date sales, as well as breakdowns for sales and orders by employee and department.

The final part of the system was administrative. Managers needed the ability to create and delete clerks, update any clerk's purchase orders, change passwordsf and override purchasing limits.

This system was put into place in December of 1999, and I was on-site to support it as it went live January 1st, 2000. I worked to fix the last minute problems and integrate it into the existing cash registers and systems. I continued maintaining this software on a contract basis for a further 7 years (until 2007).

Petrasoft also took on a second contract, a piece of software named ‘Habanero’, which is a framework to allow companies to collaborate over the internet. A new company was formed and funded to develop this software, and the new company (Lightsocket, Inc) offered all of the Petrasoft employees positions. The owner closed Petrasoft in January of 2000. I was offered and accepted a position with Lightsocket.

Software Engineer, Lightsocket, Inc
Location: San Jose, California
Dates: January 2000 - October 2000
Reported To: Gregory Stone, Vice President of Engineering
Skills: Java, Swing, Ethernet & Telephony Wiring, Windows 2000 Administration, Linux Administration (DNS, apache, users, source control systems)

I worked at Lightsocket from January 2000 until October of 2000. Lightsocket was formed and funded to develop a commercial version of software that was developed at the University of Illinois at Urbana-Champaign as a research project. This project was designed to be a framework for building collaborative tools over the internet. Lightsocket was designing, developing and marketing this software as a ‘hosted platform’ - programs that ran on our own servers.

I joined Lightsocket as the fifth employee and first engineer. I spent the first 5 weeks at Lightsocket doing physical wiring, running CAT-5 cable to each office and 12 cubicles, wiring data and phone jacks for each. I also researched server machines for our office, placed the orders for those machines, and set them up.

I was the system administrator for Lightsocket during the first few months. I was responsible for setting up a Windows 2000 domain server, and administering all the office users (which grew to over 30 before we hired a proper system administrator.) I also set up two Red Hat Linux servers to act as mail relays (using sendmail) and Domain Name Service(DNS) servers for our corporate intraweb (using Bind). I also installed and configured software packages for source control systems(CVS), File Transfer Protocol(FTP) and Secure Shell(SSH) access.

I was also the primary UI programmer for our collaboration application. I was responsible for creating all the UI frameworks in Java & Swing for the collaborative applications to run inside of. I had to travel to Illinois to work with our remote team several times to integrate the server and client components together.

During this time, I was learning a lot about working within a large engineering team and the distribution of responsibility. Responsibilities were often overlapping and it fell to the engineers to figure out what each would be doing. In a larger or more mature company, a dedicated project manager or engineering manager would have that responsibility, but due to the startup nature of our company, everyone took a hand in these tasks.

We built a first generation of the framework with several collaborative applications, including a whiteboard, a movie player, a Virtual Reality Modeling Language (VRML) viewer, a messaging system and a collaborative web browser.

I took 3 weeks of vacation in August/September of 2000 to get married and honeymoon. The week after I returned, Lightsocket was unable to secure its next round of funding to continue operations. Lightsocket laid off 2/3rds of its workers, including nearly every non-engineer. Two weeks later, Lightsocket closed its doors permanently.

Senior Software Engineer/UI Architect, ePatterns
Location: Palo Alto, California
Dates: November 2000 - May 2001
Reported to: Curtis Cole, Director of Engineering
Skills: Java, Swing, UML, XML, Enterprise Java Beans (EJB), Orion J2EE Platform, Weblogic J2EE Platform, Object Oriented Design

In October of 2000, I interviewed for an open position at ePatterns, a small software startup in Palo Alto, and was offered a position as a Senior Software Engineer, which I accepted. EPatterns was a company formed in early 2000 by several Stanford graduate students and professors to do Complex Event Pattern processing.

I was hired into ePatterns' UI team. My job was to create a User Interface to present the status of the system and display the events flowing through the system. Coming up to speed on the existing codebase and joining this team was very challenging. All members of the existing team had done postgraduate work in Computer Science, while I did not have the same vocabulary or theoretical background. I spent a lot of time reading Computer Science texts online and learning the CS theory that I hadn't been exposed to.

I learned a lot of design patterns and Object Oriented Design(OOD) - how to think abstractly about systems and programming. I also learned a lot about Application Programming Interface(API) design. Learning to design an interface that will be used by multiple systems is a delicate and precise practice, and I was part of a larger team that designed all the interfaces between different services, and between those services and the UI.

My primary task was to create a prototype UI for the first set of services that the server team had written. These servers took in a set of 'events' (an abstract term) and performed a set of analyses on those events. I created a prototype application that allowed the server team to view and modify their servers, as well as the objects and adapters in their servers.

In March of 2000, the management team was unable to secure a second round of funding, and in a cost-saving move, laid off more than half the company, keeping mostly key engineers. As part of this rearrangement, I was promoted from Senior Software Engineer to UI Architect, and was made the technical lead of the UI sections of the project.

We continued to develop an alpha version of our project, but the company ran out of money in May of 2000. The entire company was closed.

Senior Software Engineer, Netscreen Technologies
Location: Sunnyvale, California
Dates: May 2001- March 2004
Reported to: Les Arrow, Manager of Global Pro
Skills: Java, Swing, RMI, Dynamic UI Generation, TCP/IP, Firewall Administration, Object Oriented Design, Technical writing, OSI Model

I worked at Netscreen from May 2001 - March of 2004, in the Global Pro group. Netscreen is a manufacturer of firewalls, and the Global Pro group was developing a management platform to control many firewalls at once.

When I started, the group was small, 4 engineers and a manager, and my task was to learn a subset of the firewall functionality, and translate that into a set of user interfaces for the Global Pro application. I was tasked with learning how the firewall managed it's high-availability features, as well as it's DNS server, and creating UI to manage those features.

The application was broken into two parts, a UI application and a server application. The two applications used Java's Remote Method Invocation(RMI) to communicate with each other. For every UI piece I created, I had to create the appropriate RMI calls and design the API with our server engineer to ensure that the correct data was passed through.

I also acted as the in-house Swing guru, and assisted other engineers with any difficulties, as well as integrating a help system (JavaHelp) into the application.

In 2001, we released Global Pro 1.0, followed by 2.0 and 3.0 over the next two years. I was responsible for larger portions of the software as time went on, including writing specifications for other developers, and reviewing and refactoring other engineers' code for consistency and to ensure it adhered to our group's code standards.

In 2002, Netscreen acquired a company called OneSecure. Our group merged with OneSecure’s UI group, and we started to work on a brand new project that was to be the next generation of our Network Management product.

In 2003, our group worked on developing a new product: Netscreen Security Manager(NSM). This product was designed to encompass all the functionality of Netscreen’s previous product, Global Pro, as well as managing all OneSecure devices. My role in this new project was to be the technical lead for two major features. Netscreen Redundancy Protocol(NSRP) configuration, and DNS and Dynamic Host Configuration Protocol(DHCP) configuration.

This product, although built in Java, was entirely different than Global Pro. NSM was designed to have a dynamic UI, generated with a custom built framework. We designed a protocol for laying out UI elements and screens in a custom text format similar to JSON. These were called ‘views’. For each view, we also defined a set of data mappings for the data, and the framework would map the data from the views down to the database. This is very similar to how current Object Relational Mappings(ORM) like Hibernate work.

I was responsible for defining the data structures and views for each of my features, and each feature involved 10-20 structures and screens. I also created and debugged problems with Swing components that were used by our framework to actually render the UI.

I spent the next few months finding and fixing bugs for our first Feature Pack(FP) release of our software.

In February of 2004, Netscreen was bought by Juniper Networks, and between February and March, we worked on integrating the two companies.

I was responsible for writing documentation for all features I had been responsible for, as well as analyzing and documenting code that had been written or worked on by engineers who had left the company. In April 2004 we formally joined Juniper.

Senior Software Engineer, Juniper Networks
Location: Sunnyvale, California
Dates: April 2004 - May 2005
Reported to: Jens Schmidt, UI Manager
Skills: Java, Swing, C, Packet Sniffing, UML, Firewall & Router administration, Object Oriented Design

I was a senior software engineer at Juniper networks from April 2004-May 2005. I was responsible for a large new feature in our project (now renamed Network and Security Manager) - Packet Inspection.

One type of internet appliance that Juniper produced was an Intrusion Detection and Prevention(IDP) device. This device would capture suspicious packets flowing through a network and save them to a special server. I was responsible for adding a feature to our NSM software that would retrieve these packets and display them in different fashions.

These packets came in a standard format: packet capture (pcap). Although a widely accepted standard, there is no java library for reading or parsing these packets. I was responsible for writing both a library for this, and for creating all the UI to display it.

During this process, I learned about the different layers of the OSI/TCP model, how to identify each layer of the packet (Internet Protocol(IP), Transmission Control Protocol(TCP), User Datagram Protocol(UDP)) and identifying several application protocols like Hypertext Transmission Protocol (HTTP) and File Transfer Protocol (FTP). Each packet would have to be read and parsed, and the valid information pulled out. This information would include source and destination Media Access Control (MAC) addresses, IP addresses, ports and application information, as well as the protocol being used (TCP or UDP). This list of packets would be displayed and the user could interact with them in varying ways, and sorted, based upon any of the content.

I was also responsible for maintaining my existing features, fixing bugs and expanding the data descriptions and views when new features were introduced into the firewalls.

In early 2004, I was approached by a colleague who was doing some contract writing for Syngress Press. They needed an additional person to act as a technical editor for an upcoming book: Configuring Netscreen Firewalls. I signed on as a contractor with Syngress Press to do technical editing and minor rewrites. Over the next few months, I acted as technical editor for 8 full chapters and wrote 5 additional sections that were under a page each. I was also asked to write the foreword when the previous author was unable to fulfill his commitment. The book was published December 1, 2004.

At this time, I wanted to move into a more senior role designing whole projects, so I interviewed internally for a promotion as a Software Architect. Since we were undergoing a management change, a hiring decision for this position was put on hold until the change was complete. The delay extended past 3 months, and in the meantime, I was approached by Packeteer, who had an open architect position.

User Interface Architect, Packeteer
Location: Cupertino, California
Dates: May 2005 - May 2006
Reported To: Sylvia Lowden, UI Group Manager
Skills: Java, Swing, JAXB, Fedora Core Linux, Ant, Postgres DB

I worked at Packeteer from May 2005 until May 2006, in the User Interface group. Packeteer was a manufacturer of Internet appliances that performed Deep Inspection(DI) and Quality Of Service(QOS) bandwidth monitoring and shaping. The group I was hired into was designing and building a new product to replace the company's 8 year old management software.

I was hired into a group that consisted of a manager, two User Experience(UX) designers, and one junior engineer. I was given the position of UI Architect, and was expected to take all the designs and workflows created by the UX designers and code it.

The UX designers followed a process called Interaction Design, primarily created by Alan Cooper and Cooper Design. I attended a week long class in-house, that introduced many of the concepts that we were using. I spent the first few months working along side the UX designers doing paper prototyping, visual design, workflow design and UI design. We used several tools to document these, including OmniFocus and UML.

I worked directly with the server architect to design service API's and data objects for use when transmitting data between the server and the client. We used JAXB(Java Architecture for XML Building) to both generate our data classes, and to marshall and unmarshall data on each side. Our server platform was deployed on Fedora Core Linux, with our data being stored using the Hibernate ORM, and persisted in a Postgres DB. I managed several server devices, which included setting up Linux, installing and setting up Postgres, and managing and extending developer tools such as Ant and Apache Axis.

At this time, I was also designing and programming a UI framework for dynamic UI generation that would allow a programmer to ‘lay out’ a UI screen or ‘view’ in a simple XML format and render the views, while mapping each field to a data object. I implemented the entire framework and coded it in Java and Swing.

I was also given a job requisite to fill for a second engineer on my team. I worked with our technical recruiter to create and post an appropriate job description. I screened candidates and was the primary technical interviewer for candidates. I was the final decision maker on which candidate would be hired.

I was responsible for overseeing both junior programmers on a day to day basis. I documented and delegated tasks, tracked their progress, and mentored them both on our technical processes and on how to manage their time.

Our team produced an alpha product that tested very well amongst our users. Due to a senior management change, our group was disbanded and our project shelved. Instead of attempting to find a different position inside the company, I decided to leave, and was able to secure a position at Apple.

Senior Software Engineer, .Mac & MobileMe Groups, Apple Computers
Location: Cupertino, California
Dates: May 2006 - May 2008
Reported To: Jake Baumgarten, Back End Server Engineering Manager
Skills: Java, Swing, HTTP, WebDAV, Web Services, Multithreaded Programming, Apache Tomcat, XML

I worked at Apple in the .Mac Back end server engineering (BaSE) group from May 2006 until Mid 2007. The BaSE group was responsible for two major internet services: synchronization and storage. I was primarily working on the storage aspect, but was often ‘on loan’ to the synchronization team.

The storage group was responsible for a set of software running .Mac’s “iDisk” service - a remote storage service that used the Web Distributed Authoring and Versioning(WebDAV) protocol to communicate with MacOS X and other clients.

I had very little experience with writing servers, so I spent some time learning the foundations that the servers had been built on, which was a modified version of Apache Tomcat. The protocols were WebDAV which is an extension of the standard Hypertext Transfer Protocol(HTTP). I read the specifications(RFC’s) and developed a working knowledge of these systems and protocols that I could build on.

My first task was to decouple our storage from a disk-based system and make the persistence layer ‘pluggable’. I created several backing store implementations for this, including the existing disk-based storage, a remote WebDAV backing store (that would write and read the data to any remote WebDAV server), and an in-memory based implementation for testing.

My second task was to work with our operations group to remove a set of proxy machines and replace them with a hardware load balancer. During this process, I was required to learn about our entire network architecture and routing infrastructure. Working with a small group, we broke our machines into separate banks and classified each bank for a specific type of internet traffic.

My next task was to re-write an in-house application that we used as a test harness. Our group used this application to run regression tests against our varying servers. This application was using outdated toolkits and was very difficult to modify. I rewrote the entire UI in Swing, and added a programmatic Command Line Interface(CLI). I worked with our build team to integrate it into our continuous build system, so that whenever changes were made in our servers, the appropriate test suites would run automatically, and the results emailed to the engineering group.

I was then put in charge of finding a better solution for Apple's iWeb publishing feature. Apple's iWeb software was able to create and publish web sites to the .Mac web service, but the publishing solution was buggy and inconsistent. My boss and I designed a new publishing protocol. This protocol involved staging assets to a temporary server location and a special server operation that could walk existing published directory hierarchies and overlay new or modified resources. This set of operations allowed there to be a consistent view of the web assets. I coded a large portion of this feature, which required a lot of refactoring and modifications to our existing code to deal with temporary locations, aging out files and resources, and cleaning them up periodically. I also designed and coded a background task processor that would handle these cleanup tasks.

My next job was to work with another server engineer to create the Application Service Interface(ASI) and write the code for .Mac Gallery - a new feature for publishing photos from iPhoto to the .Mac service. I wrote the ASI for publishing and retrieving the images and albums, as extensions to our WebDAV protocol. I also integrated a service for resizing photos on-the-fly, based on requests from the browser. These requests ran through a separate image processor that was Objective C based.

Throughout the middle and end of 2007 we began work on new features for our upcoming rebranding and release: MobileMe.

Senior Software Engineer, MobileMe Group, Apple Inc
Location: Cupertino, California
Dates: Late 2007 - April 2010
Reported To: Jake Baumgarten, Back End Server Engineering Manager
Skills: Java, HTTP, WebDAV, Web Services, Multithreaded Programming, Apache Tomcat, XML, REST, BerkeleyDB

By the third quarter of 2007, I was fully working on Apple's rebranding of our web services, to be called MobileMe.

I was temporarily assigned to the synchronization team who were rewriting their entire service. I was given the task of creating a set of utilities to provide locking and atomic operations for their multithreaded server applications. The new sync servers were built on top of our storage platform, but needed these extensions. I resolved over a dozen high priority, critical issues to unblock their team and help them meet their deadlines.

My next job was to write new Application Service Interfaces(ASI's) for the upcoming MobileMe release. We were providing web interfaces to our iDisk and Gallery functions, and I was put in charge of exposing the protocols for these. We moved to using a new protocol, a Representational State Transfer(REST) based-protocol over HTTP.

The iDisk Web Service required a basic set of filesystem semantics - the ability to send, receive and edit files, as well as the locking and metadata semantics all needed to be exposed. I designed wrapper layers for exposing all these via a REST-ful web service. I also created new functionality to create archives (.zip files) on arbitrary collections of files, and exposed this functionality via a web service.

My next task was to create a web service for manipulating the .Mac gallery - adding, modifying and deleting photos and albums. This new RESTful protocol was much more powerful and terse than the previous WebDAV based protocol, and we petitioned to have all current clients (including iPhoto, iMovie and QuickTime) to transition to using it.

I wrote technical specifications describing the new Gallery Publishing Protocol and became our team’s primary liaison to these client application teams. I would frequently work side by side with their engineers, looking at TCP traces and debugging the protocol or other difficulties during integration.

My next job was to work with another newer engineer on our team, and bring him up to speed on the Gallery Publishing protocol, code and server configurations. Once he had been trained, I turned over primary maintenance of this feature to him.

Part of our architecture for tracking assets in a user’s gallery was stored on a set of disk appliances using BerkeleyDB. We had a heavily customized version of BerkeleyDB, and due to performance issues, I had to modify the BerkeleyDB source code to allow a multiple reader semaphore or single reader/writer mutex locking system. I worked closely with a database expert and made all these code changes. I then worked with our Quality Assurance team to create a test matrix to catch any unexpected error conditions.

In early 2010 I was recruited to a project that would replace MobileMe.

Software Architect, iCloud, Apple Inc
Location: Cupertino, California
Dates: April 2010 - July 2013
Reported To: Patrick Gates, Director of Engineering, Jake Baumgarten, Cloud Services Manager
Skills: Java, HTTP, Multithreaded Programming, XML, Web Services, NoSQL Databases, Oracle, CardDAV, WebDAV, Python, Memcache, Google Protocol Buffers

I worked in Apple's iCloud division from April 2010 to July 2013. When I joined the team, I reported directly to the Director of Engineering. In early 2011 the team was reorganized and I joined the newly formed DAV (Distributed Authoring and Versioning) team.

As this team was brought together to create an entire new large-scale web service, I did a great deal of research. I spent several months researching libraries and frameworks for web containers, multithreaded and non-blocking input/output. Some of the technologies that I evaluated were Netty, Jetty, Mina, and Tomcat.

I did research into databases, persistence layers and NoSQL frameworks. I evaluated MongoDB, Cassandra, Oracle, Hibernate, Cages and Zookeeper.

I was put in charge of building an HTTP and base WebDAV server on top of a multithreaded and asychronous I/O(Input/Output) framework. This server later became the platform that iCloud would build its Personal Information Management(PIM) servers upon. This project involved doing programming work to store and retrieve both content and metadata for web resources. I converted an older integration test harness to exercise our new servers, and wrote and converted over one hundred tests to exercise base HTTP and WebDAV functionality.

I was also responsible for defining a new protocol for the storage, retrieval and synchronization of browser bookmarks across multiple devices. I worked with a client engineer and wrote a complete RFC defining this protocol.

I forked our base server and created the first implementation of a bookmark protocol server. I coded the alpha and beta versions of this server, and wrote the test cases. I also trained two other engineers on our protocol and code base.

I was then put in charge of a second server that had been built on our base, a CardDAV server, designed to implement RFC 2426. I inherited a half-complete codebase with an incomplete test suite, and was expected to turn it into a complete scalable web service in ten months time.

I started by examining the test harness. The harness was written in Python, which I had little experience in, so I dug into it and started learning. Within a few weeks I had a decent knowledge of both the CardDAV protocol, and the test harness. I used this knowledge to expand the suite of tests to over 100 integration tests, and several dozen unit tests.

My next job was to write a set of tools to store binary data to an external source. I wrote a set of wrapper classes and utilities that allowed any persistent data store to be used, and switched at will.

Next I refactored all of the server code and removed places where we had duplicate code across our servers. I moved common code into our base server where all the other servers could benefit from it.

My next job was to re-write the lexical parser for our data. The vCard RFC defines a specific set of requirements for handling vCard data, and we had been using an open-source library that was poorly written. I eliminated it and wrote a complete vCard parsing library as a separate project. The new library was over 500% faster and was less than 30% of the size of the original.

I spent time writing a performance test matrix to measure our capacity for individual users, individual servers, databases (connections and throughput), and for banks of servers. I worked closely with our performance team to create performance test suites and debugging problems so we could evaluate changes to our system quickly.

During this time I learned a great deal about market pressures and evaluating tradeoffs in time and quality. I became adept at gathering data to make an informed decision about the impact of varying bugs or decisions. I learned that having data to back up a decision is absolutely necessary.

I worked closely with multiple groups and stakeholders while developing iCloud. I learned how to balance multiple requests and tight deadlines while under pressure. I was often the lone representative from our group while interacting with other application groups, web service groups, product management and marketing.

iCloud finally launched in October of 2011. Within the first 100 days, we had over 100 million customers. I spent the first few weeks coping with our launch and unexpected fallout. I then spent several weeks working with customer support to analyze and solve customer issues. I was always on call for high-profile cases in the event that an Apple Executive escalated a difficult issue that was related to my area of responsibility.

In 2013 I was approached and asked to consider moving to a different group internally, a group that needed an experienced architect to help with scaling and development of new features.

Software Architect, iCloud, IDS & FaceTime Apple Inc
Location: Cupertino, California
Dates: July 2013 - Present
Reported To: Andrew Vyrros, IDS Manager
Skills: Java, HTTP, Multithreaded Programming, XML, Web Services, NoSQL Databases, Python, Memcache, Google Protocol Buffers, Protocol Design

I moved over to the IDS & FaceTime group in July 2013 and have been doing Research & Prototyping for new projects which have not yet been released, in addition to re-writing some existing services to provide more stability and scaling.

I hope to work at Apple for many years, because I am constantly being challenged by new technology and projects, and I realize how lucky I am to be at a company that is changing the world.

Courses Petitioned:
Software Engineering 1 - CS 305
Software Engineering 2 - CS 310

Department: Information Technology & Engineering

Credit Hours Requested: 6 Hours (Upper-Division)

Course Description: This course provides a broad introduction to software engineering theories, methods, and tools. Topics include requirements engineering, analysis and design, implementation, versioning, and testing.

In 2000, I worked as a Senior Software Engineer at ePatterns, in Palo Alto, California. I reported to Curtis Cole, the Director Of Engineering. I began as a Senior Software Engineer, working on a small, 3 person User Interface(UI) team. The company was formed by several Stanford graduate students and professors to create a business application for Complex Event Processing (CEP). The software was intended to take in varying 'events' through multiple sources - legacy databases, manual input or system events, and apply a set of processing rules. The central part of this system was a Rules Engine (RE). The Rules Engine was essentially a named-edge graph that could have arbitrary processes applied to data that travelled the edge(or path). A definition could be made for each input adapter, data transformer and output adapter in our domain language (A rudimentary extention of XML), and the rules engine would parse that definition and process the events as defined.

This system required a lot of learning on my part - I was the only member of the team that didn't have postgraduate education in either Math or CS. The way the system was defined caused us to have a very rigid Application Programming Interface (API) for the adapters and the transformers. At ePatterns, we had a small team of people who were responsible for designing and reviewing all API's for the system. I was asked to be on the team, and this required me to learn a great deal about OO design - designing interfaces, utilizing base classes and proper composition and inheritance of the code.

I also managed to learn a great deal about data structures like graphs and trees - we used a named-edge object graph as the base data structure for our rules engine, and had several tree structures for finding the correct transforms inside the transformers.

From 2001 to 2004, I worked as a Senior Software Engineer for Netscreen, Inc in Sunnyvale, CA. I reported to Les Arrow, and was one of the first members on a team to develop a brand new piece of software called GlobalPRO. This was my first introduction to the System Development Life Cycle (SDLC). Our team was required to take a set of functionality that already existed on Netscreen Firewalls, and create a piece of software to manage multiple firewalls. I was responsible for several pieces of this software, including representing and managing the High Availability and DNS server portions of the firewalls. I had to analyze the current UI the firewall had, determine the low level commands, and work with the firewall engineers to fully understand the requirements. I worked with our team's Project Manager to determine timelines and estimates for the required work, and wrote feature specifications. Next, I would design the UI and outline the code using UML. I would review the design, and then implement all the code. I would write unit tests to test specific individual features, as well as write a test plan for end to end testing. I would test all of my features on multiple devices (Netscreen produced 5 different models of firewall, all of which were managed by our software). I would then be responsible for maintaining the software and responding to bug reports, as well as outlining improvements and refactoring the code for the next version. As our team grew, I was also one of a three man team that was responsible for reviewing all the major portions of other engineer's code to ensure that it met our 'best practices', which included proper design patterns such as: using dispatch threads so as not to block the Swing event thread in Java, efficient client-server communication, as well as using proper synchronization between the server and client.

In 2004 and 2005 I worked on the replacement for GlobalPRO, which became known as Netscreen Security Manager, and later Juniper's Network and Security Manager (NSM). This was a piece of software that was easier to expand to deal with the newer and more disparate pieces of hardware that Juniper was producing. We went through a very similar cycle as before, writing a full piece of software from scratch. I was also involved in a few pieces of research for the new software, the most interesting of which was deciding whether to use a custom protocol for client/server communications, or to use an established protocol like SOAP or HTTP. A part of this investigation was learning about the different encryption methods that are commonly used, like SSL and TLS, and the difficulty in implementing these in raw C and Java.

In 2006 I began work as a Senior Software Engineer at Apple Inc, in Cupertino, CA. I worked for Jake Baumgarten in the .Mac (and later MobileMe) groups. This was the first time that I had worked on a huge distributed system, and again it required me to stretch my knowledge and learn a lot. The distributed server system that we worked on was a large web service, based on a highly customized version of Apache Tomcat, and optimized for large sized transactions. The requirements for this system were pretty strict, as we had many millions of users. All code that was committed had to conform to industry standard best practices. This included proper object oriented design, class and package breakouts, clean and well documented API and SPI design, and tight encapsulation of each module. Every line of code was rigorously reviewed and refactored before going to our test and production systems. I learned the value of planning for the future, as all of our software and persistence layers had to be backwards compatible at all times - our servers had to be upgraded a few at a time to ensure that our service was never ‘down’. For over 2 years, we never had a full outage. I also learned the value of rigorous performance testing - as the smallest deadlock or poorly coded mutex could easily cause a hundred threads to back up behind it. During this time, I was the technical designer and implementor of an entirely new feature that turned out to be one of the services most popular: .Mac gallery. This required me to create multiple outward facing API’s for mobile use as well as desktop applications. In 2009, I was promoted to the MobileMe platform architect, where I became responsible for overseeing all the core frameworks and platform code that the entire service was built on. I also produced several full features, including a complete rewrite of our web and mobile API's to be a more 'RESTful' type of web-based API (REST stands for REpresentational State Transfer, and is an architecture for web systems)

In 2010 I was recruited to work on the replacement for MobileMe: iCloud. Here again, we had to design a huge system from the ground up. Our team was given a set of requirements by a committee, and our small group was given the entire creation and SDLC task. We looked at the basic requirements and started analyzing the types of systems we would need. My primary responsibilities were the Personal Information Managment (PIM) applications, and I also worked as part of the core platform team. The core platform team reviewed different frameworks and persistence layers, evaluated their performance and ease of use, and presented these to the director. Once a choice had been made, I worked with a client engineer to write an RFC for the synchronization of browser bookmarks, based on similar technologies and RFC's ie: Calendar Syncing (CalDAV), Contacts syncing(CardDAV) and the XML Bookmarks Exchange Langage (XBEL). We completed the spec, and I designed and implemented the first version of the protocol server. I then moved on to the Contacts PIM/CardDAV server, which had only a partial prototype. I rewrote the entire thing, and have been the primary coder and maintainer on both the CardDAV protocol server and several core iCloud platform frameworks since then. I have just recently handed this role off to two other engineers, and have moved to a new group, where I am doing research, prototypes, and designing new features for the future of iCloud & iOS.

Course Petitioned: Internetworking - CS 320

Department: Information Technology & Engineering

Credit Hours Requested: 3 Hours (Upper-Division)

Course Description: Principles and issues in interconnecting multiple physical networks into a coordinated system, operation of Internet protocols in the interconnected environment, and design of applications to operate in this environment.

Around 1989 I became interested in computer networking - I owned a 1200bps modem that I used on my Commodore 128. It was a hayes-command compatible modem that supported a basic XModem protocol - I used it to dial into different BBS systems, and even ran my own.

In 1992, I had upgraded to an IBM compatible machine, and I was introduced to the fledgling internet via a dial-up account at the University Of Windsor. This provided me a raw terminal that dialed into the university's mainframe, and was my gateway into learning how to use http, ftp, irc, gopher and a unix style shell (I was running on Windows 3.1 at the time.)

In 1999 I began work at Petrasoft, a small contract engineering company in Santa Clara, California.

I was the second employee and when we moved into our first offices, I was given the job of running Category 5 (Cat-5) cable to each office and tying it into each office for phone and ethernet. I learned about physically connecting Cat-5 and breaking out the separate twisted pairs for multiple use. In this case, we would use 2 pair for 100Base-TX fast ethernet, one pair for standard telephony, and the last pair went unused. I learned to properly wire ethernet outlets and create standard and crossover cables (for connection to non-switched hubs or other computers). Once our offices were physically wired, I was responsible for setting up our network, Windows NT 4.0 server, and three Windows workstations. Our office had a DSL line into it, and I had to learn how to network all the PC's together, as well as provide a windows domain for access to shared drives and our source control system (Microsoft Source Safe). AT&T provided a DSL modem with a single uplink and a page listing our one static IP and the upstream gateway. I spent several days with a single workstation plugged into the uplink learning about Windows networking. Through some internet help and trial and error, I was able to configure the Windows NT Server to act as our router and proxy. It had two ethernet card, one was set up with the static IP pointing at the internet, and the other was set up with an internal interface (192.168.X). I configured the server to pass traffic through, and to also vend IP addresses via DHCP to the workstations on the network. I had to learn about DHCP configuration and Windows domains. I created a domain for our users and set up access to our source control on a per-user basis. I installed and configured Windows Internet Information Services(IIS) web server for our company's web page. I modified the configurations to allow incoming HTTP requests to IIS. I configured several shared drives, and exposed access to a samba (SMB) share to share information with clients.

In 2000 I went to work for Lightsocket, a startup in San Jose, CA. Again I was the first engineer on staff, and with the help of our second engineer, I wired all of Lightsocket's offices for telephone and ethernet. I ordered our first servers and workstations and set up the office Local Area Network and our Windows 2000 domain controller. I also was asked to set up a mail server, which I had never done. I was in the process of installing RedHat Linux on a server to act as our source control and FTP server, so I spent a week learning about Sendmail configuration, and was able to successfully configure it. I maintained these machines until we hired a full time System Administrator several months later.

Lightsocket also had a high speed internet connection, and due to the number of people and differing tasks, we decided to build two different LANs - one for engineering with access to the source and build servers, and one for non-engineers. I learned about segregating subnets and subnet masks, and was able to divide the LAN's accordingly. Each subnet needed it's own DHCP range to vend correct IP's, and each subnet was connected to one ethernet interface on the Windows 2000 server. Lightsocket also had a hardware firewall, a Netscreen 5. This firewall was the point of entry into the company's network, and also provided us with several handy features. The company had 5 static IP's, and I was able to set up several Mapped IPs (MIPs). This would directly map a single external IP into a single internal IP, based on a set of rules. I mapped one to our Web server, and served all of our company's web pages from a server running Apache on Red

Hat Linux. The firewall also allowed us to configure a Virtual IP(VIP) for our main static IP. I used this functionality to map DNS, SMTP and SSH into several different internal servers. I was also responsible for setting up a VPN to our remote branch in Illinois. The group in Illinois also had a Netscreen 5 attached to their LAN. I configured their LAN as a separate subnet (192.168.3.X) and configured routing rules on the firewall to send all traffic to that subnet through the VPN. Configuring the firewall also introduced me to the concepts of Trusted and Untrusted zones.

From 2001 until 2003 I worked at Netscreen Technologies, working on their firewall management software, Global Pro. Netscreen was a networking company, and most of the engineers knew and understood the networking stack from top to bottom. I had sporadic knowledge, but in order to come up to speed on what our products did, I started reading and learning about TCP, UDP, and the OSI model. Netscreen firewalls could do routing and firewall actions anywhere between Layer 2 and Layer 7.

In 2004, Juniper Networks bought Netscreen, and I was given a new job - write a new piece of software to integrate into our newest management platform. This feature was to read packet captures that had been logged by the firewall into .pcap format, parse them, correlate them and display them. This required me to learn about the format of packets - the MAC addresses, IP addresses, and layer 4 information - either TCP or UDP data, as well as the payload. This information would then be matched with other parsed packets, and displayed in our client. The user could view the packets as a stream of communication, sorting by time, source IP, destination IP, source or destination MAC, and source or destination port.

I began work at Apple in 2006. For the first time, I was working in a server group, writing large scale web services. Our main platform was a modified version of Apache Tomcat. I spent a lot of hours learning the ins and outs of the HTTP and WebDAV[1] (Distributed Authoring and Versioning) protocols, as they were what all of our applications were built on. The group that I worked in was responsible for the two largest portions of our service: storage and synchronization. The storage aspect was represented by a WebDAV protocol implementation, providing filesystem semantics over HTTP. Using the DAV extensions, we provided metadata and locking as well as the ability to store and retrieve content over standard HTTP connections. I was part of a group that implemented user-level Access Control Lists (ACLs) based on the DAV ACL spec[2]. This was a very interesting project, and provided insight into user and group level security, and the difficulty of mapping different users into different access roles.

In 2010 I began working on iCloud, and wrote two of the iCloud web services, for contact synchronization using the CardDAV[3] protocol and a Bookmarks synchronization protocol that I developed and co-wrote with another Apple employee.

[1] http://datatracker.ietf.org/doc/rfc4918/

[2] http://www.ietf.org/rfc/rfc3744.txt

[3] http://www.rfc-editor.org/rfc/rfc6352.txt

In 2013 I moved to a new group within Apple, where I am doing research, prototypes, and designing new features for the future of iCloud & iOS.

I am unable to disclose more about my work with iCloud due to my NDA with Apple. I am still in this role today.

Work References

Examples of my code for my student project is available on my github:

https://github.com/Kylar42/ntp-scheduler

Due to the nature of my employments, I am unable to make examples of my work-related code available.