QUENTIN PIERCE

qtpierce@gmail.com 503.407.8670
www.linkedin/in/quentin-t-pierce www.qtpierce.com

Senior Hardware Design Engineer

Senior electrical engineer known for automation solutions using skills from both HW and SW domains. Examples:

- Static-code-checking enables a computer to teach a junior engineer to code with fewer bugs.
- Automated devices that push work onto the computer.

The following are the SOAR stories I use when I interview.

Hardware Engineering.

Situation - **A CPU was failing its final validation testing.**

Obstacle - The tests involved input data, output data, and golden data. And a logic analyzer. No one else in the team knew the deep problems of logic analyzers.

Actions - I explored the patterns of the failure. The failure occurred on a 1MHz frequency. I found the 1MHz DC-DC convertor was causing the false failures. I wrapped the logic analyzer probes in aluminum duct tape and the interference was resolved.

Results - The team immediately approved the passing tests and authorized the CPU for shipment.

Skills – debugging, logic analyzer usage, EMI knowledge

Summary - Used my logic analyzer skills to finish a test no one else could; I built a Faraday-Cage from duct tape.

Designed the TDS5000 thermal printer system.
Situation - Customer requirements listed a need for a printer in the TDS5000 oscilloscope. This was my first design project.

Actions - I learned to work with component engineering, to contact vendors, and to use the schematics capture tools. I selected the components and learned about them. I designed the thermal printer circuitry. I tested the thermal printer circuitry.

Results - Tektronix offered the thermal printer as an enticement to pull in customers.

Skills - Work with a loose knit team, schematics capture tools, learning about ICs,

Summary - Learned to design thermal printers, use schematics capture tools, and speaking with vendors in order to offer a thermal printer with certain oscilloscope models.

Researched trends in motherboards and championed the adoption of those trends.
Situation - I saw coworkers misunderstanding the potentials of commercial computer components.

Actions - I used a search engine to collect information on the available features in many commercial computer components. I created summaries of the information that showed trends in the features. Some features were mandatory, some features were exclusively offered by a few vendors. And some features were emerging onto the market.

Results - When I released my reports, my coworkers saw the information and began to accept ideas that used the mandatory or the emerging features. My coworkers also benefited by having attention brought to the exclusive features so that we could decisively choose to accept or reject exclusive features.

Skills - Internet data mining, report writing, product feature comparisons, presentations.

Summary - Learned about technology trends in the PC industry so that the concerns of coworkers could be addressed.

Designed a relay test board.
Situation - Component engineering needed someone to design a relay test board.

Actions - I designed the relay test board. I had a turn-key board fab house build the board.

Results - Component engineering got the board they wanted.

Skills - Board design, sharing work with another, turn-key board design.

Summary - Designed a board for Component Engineering's needs to test relay components.

Designed the DPO series interface boards and wrote specifications.
Situation - Two interface board designs were needed, one per model in the series.

Actions - Created an interface design and reused the circuitry to create the second design. The design was a low-risk design and it started from a known good design. Wrote documentation for this low-risk design. A second design was needed. A high-risk design with new components. Created a new interface design with the new components and reused the circuitry to create a fourth design. Wrote documentation for this high-risk design.

Results - Risk was removed from the software engineering schedule because the low-risk design worked correctly with obsolete components. Software engineering started writing software while I worked on the high-risk design. Both designs were reused to save development time. The documentation was accurate and presented to both engineering and manufacturing for educating people about the design.

Skills - Digital design, analog design, planning for risks mitigation, specification writing.

Summary - Created two interface board designs, reused them twice, and documented all four board designs for posterity.

Championed the sharing of the DPO series interface circuitry and sharing of the specs.
Situation - Kept diligent notes on the design of the interface circuit. Wanted to put the notes to good use and educate people.

Actions - Wrote the engineering notes into a board design specification and gave it to engineers who wanted to reuse the design. Spun off a copy and pared the copy down to basic design theory and troubleshooting notes and gave that copy to repair technicians for their education.

Results - Several engineers learned from my design spec, people told me the design spec looked great, and repair technicians had a handy document that listed real problems and solutions for the interface circuit.

Skills - Documentation, team work, sharing, teaching technicians.

Summary - Wrote excellent design specifications and shared them with engineers and technicians, allowing both personnel to become familiar with the interface circuit.

Learned Verilog: an FPGA firmware language.
Situation - The new architecture concepts required data processing and FPGAs were chosen as the processing devices.

Actions - Knew the basics of Verilog. Asked coworkers to assist in learning the NC-Verilog environment. Took a class on Verilog coding methodology. Participated in developing a Verilog coding methodology. Practiced coding Verilog code and test code. Developed two pieces of FPGA firmware using Verilog.

Results - Learned Verilog coding well and wrote firmware and tested firmware.

Skills - Verilog, architectural planning, FPGA synthesis.

Summary - Strengthened Verilog coding skill set and wrote the necessary firmware and synthesized an FPGA to fit a need for a new piece of hardware.

Worked with other divisions to review their interface circuit designs.
Situation - Different divisions required circuit reviews.

Actions - Worked with oscilloscope divisions to review their interface circuit designs. Worked with logic analyzer and signal source divisions to review their interface circuit designs.

Results - Different divisions got their design review efforts.

Skills - Design review, learning from others.

Summary - Worked with the many Tektronix divisions to perform design reviews on their circuitry.

Worked with mechanical engineers to understand thermal problems with Intel CPUs.
Situation - Mechanical engineering was concerned over the TPD of modern Intel CPUs.

Actions - I established a CPU thermal stress test. I used Intel's CPU thermal specification as the starting point and established a recommended system level thermal specification. I verified my system thermal specification on two different models of oscilloscopes.

Results - I handed mechanical engineering a report on expected CPU behavior across temperature points. I handed mechanical engineering a recommendation on ambient thermal specs and system airflow.

Skills - Writing a CPU stress, Intel's CPU thermal specs

Summary - Worked with mechanical engineering to derive thermal specifications for complex PC systems.

Worked with quality engineers to understand reliability problems and failure mechanisms.
Situation - Manufacturing group asked for engineering to improve the reliability of an oscilloscope model.

Actions - I provided information and ideas concerning how to find failure mechanisms. I provided insight into dealing with failure mechanisms. I supported the team's efforts.

Results - Reliability improved %10 because of the team's efforts. Reliability improved some unknown percentage because of my efforts. I learned concepts in reliability.

Skills - Fault analysis, contacting sub-contractors, service policy.

Summary - Worked with a team to improve the reliability of oscilloscopes.

Worked with software engineering to develop drivers for hardware.
Situation - Software engineering needed a hardware engineer to help develop additional functionality into the hardware driver.

Actions - Worked with three software engineers to develop functions that setup, controlled, and handled interrupts for a hardware system that was built into an FPGA. Worked with software engineers to correct design flaws in the hardware driver. Helped the software engineers understand the timing and state needs of the hardware.

Results - The additional hardware system in the FPGA worked. Several bugs in the driver were located, corrected, and documented.

Skills - C++, knowledge of the system architecture, software debug concepts.

Summary - Worked with software engineers to develop and debug hardware drivers. Taught the software engineers knowledge of the system architecture. Learned from them how to debug software.

Designed mixed signal circuits.
Situation - Marketing wanted a thermal printer for an oscilloscope.

Actions - Designed two thermal printer systems. One was a complete design that used a USB bridge to control a thermal printer head and microcontroller. The second was an interface and power supply for an off-the-shelf monolithic thermal printer. Thermal printers are a mixture of digital design, analog design, power system design, and mechanical control. Analog sensors and digital sensors are used to update the microcontroller of the state of the thermal printer.

Results - Two models of oscilloscopes have thermal printers in them that work. I learned some lessons on mechanical control.

Skills - Digital design, analog design, machine control, driver debugging.

Summary - Designed two thermal printer systems and learned mechanical control, mixed signal design, and power supply design.

Learned industry standard buses.
Situation - Interface boards use industry standard buses to connect oscilloscopes into PC components and PC processor units.

Actions - Learned to design, debug, and interpret the PCI bus, USB, RS232, and parallel port buses. Designed the PCI interface in the latest Tektronix oscilloscope models. Worked with other divisions to reuse that design, complying with management's request for more design reuse. Designed much of the USB interface in the latest Tektronix oscilloscope models. Worked with Ethernet as well.

Results - Two models were built with the same PCI interface design. Two other divisions copied the design. This saved development time on three models of test equipment. And saved repair and debug time on those models as well because all models shared the same problems and solutions.

Skills - PCI & USB architecture, RS232 signaling, PCI & USB circuit layout, Ethernet signaling.

Summary - Developed an interface system using PCI, USB, Ethernet, and RS232; the system was reused three more times to speed development of other test equipment.

Learned about high-speed ICs.
Situation - High-speed circuit and IC design occurred all around me.

Actions - Participated in many discussions and read material concerning high-speed circuit designs and high-speed IC designs. Participated in qualification and debug testing of high-speed circuits.

Results - Learned terminology, failure mechanisms, and troubleshooting techniques for high-speed systems.

Skills - Terminology, test methods, concepts.

Summary - Exposed to high-speed design theories and participated in debug and verification testing of other people's high-speed designs.

Designed with LVDS and CMOS differential signaling.
Situation - Did not know anything about differential signals.

Actions - Learned through USB designs about CMOS differential signaling. Learned through FPGA designs about LVDS signaling. Probed the signals and wrote documentation for repair technicians to ease their understanding.

Results - Gained an understanding of differential signaling.

Skills - Signal probing, LVDS design, differential design.

Summary - Used LVDS and differential signaling to connect circuit systems together. Used LVDS in two FPGA designs.

Familiar with modern logic signal types.
Situation - Did not know anything about logic signals.

Actions - Learned TTL, CMOS, ECL, and low voltage variants. Designed with those modern logic signal types.

Results - Gained an understanding of modern logic signal types.

Skills - Signal probing, bus design, single-ended logic types.

Summary - Used modern single-ended signals to connect circuit systems together.

Familiar with high-speed analog-to-digital conversion.
Situation - Did not know anything about high-speed analog-to-digital conversion.

Actions - Learned about high-speed analog-to-digital conversion from people who are leaders in the field. Participated in design reviews and verification testing of the AtoD systems of oscilloscopes.

Results - Learned about analog-to-digital conversion and oscilloscopes from industry leaders.

Skills - AtoD design theory, oscilloscope usage

Summary - Studied high-speed analog-to-digital conversion with industry leaders.

Familiar with low-speed digital-to-analog conversion.
Situation - Did not know anything about low-speed digital-to-analog conversion.

Actions - Learned about digital-to-analog conversion by understanding the calibration system of an oscilloscope; DtoA is used throughout the calibration system of an oscilloscope.

Results - Learned about digital-to-analog conversion and about calibration systems of systems.

Skills - DtoA design theory, calibration theory

Summary - Studied digital-to-analog conversion by understanding the calibration system of an oscilloscope.

Experience in writing work instructions and engineering revisions.
Situation - Work instructions/engineering revisions needed to be written.

Actions - I have written work instructions and engineering revisions. Engineering revisions were ISO9000 compliant. The work instructions were the precise details necessary for the technicians and assemblers. How to program the PAI device, how to install computer hardware into scopes, how to repair/modify/test circuitry.

Results - Coworkers were happy to receive clear instructions on how they were to perform some tasks.

Skills - Work instructions, engineering revisions, engineering change requests

Summary - Wrote work instructions and ECRs well enough to convey the information to peers, managers, and technicians.

Experience in computer system failures.
Situation - I see several trends/patterns in the computer failures I am told about/involved in.

Actions - I maintain my own computer network, as well as computers belonging to friends and family. Coworkers know this and come to me for advice on computer repair. I decided to document computer failures I am personally involved with.

Results - I have a pair of documents I keep updating. One details why computers fail to turn on. Another details hard drive failures.

Skills - Documenting and researching failure mechanisms, experience in computer failures

Summary - Developed documentation and experience about computer failure mechanisms.

Software Engineering.

Situation - **Writing RTL code is hard and precise work.**

Obstacle - How do we enforce coding standards on the new engineers?

Actions - Write static-check lint rules that read RTL code and look for violations of the coding standards.

Results - I've had junior engineers thank me for teaching them to run my static-check tool because it taught them how to adhere to the coding standards.

Skills – Perl, C++, code linting, RTL coding.

Summary - Wrote static-check lint rules that helped junior engineers improve their coding quality.

Situation - **I did not know how end users would use my SW. I suspected there maybe important lessons in knowing.**

Obstacle - There was no training concerning how end users would use our SW. And all the end users were too busy to teach me.

Actions - I asked for a half time project during the seasonal down time in my workload. The team I joined had never used my SW. I taught them how to use my SW which helped them discover design flaws. I also asked for simple RTL coding work from them as well.

Results - Teaching them gave me an opportunity to rewrite documentation. Writing simple RTL code for them gave me an opportunity to find some pre-silicon bugs in their design.

Skills – managing tasks, cross-training, helping others, RTL coding.

Summary - Cross-trained with a customer in order to learn how they use the SW.

Situation - **A static check rule took 3 hours to run, which was too long.**

Obstacle - I knew the problem code was written as a long process that "walked a tree and asked 12 questions" at every node in the tree.

Actions - I used my principle engineer to understand the underlying problem the code was checking. I rewrote this portion of code, changing it from "walking a tree and asking 12 questions" into "walking the tree 3 times and asking only 1 question of each node. Once the known-bad nodes were found, the code then "asked 12 questions of the known bad nodes".

Results - The 3 hour task turned into 15 minutes. CPU designers were delighted and began using that test more frequently. The test located broken wiring inside the CPU, so not running the test led to broken CPUs.

Skills – Perl, algorithm design, refactoring, RTL coding, code linting.

Summary - Learned the background principles of a problem and refactored an algorithm to speed it up.

Situation - **A data indicators package took 3 days to upload a dataset; so no one liked using it.**

Obstacle - No one knew why it took 3 days. I explored the indicators package code and found it did not perform bulk-uploads.

Actions - I then took ownership of the indicators package, negotiated with the database team for a a bulk-upload feature, and added calls to the bulk-upload feature. I then automated the process, documented everything in a wiki, and added some robustness to it.

Results - The data indication step now takes 10 minutes to upload. It now runs daily and survives crashes coming from the database team's SW.

Skills – MySQL, delegation, automation.

Summary - Owned a set of process indicators and turned it into a useful, automatic, process.

Situation - **End users did not see proof of bug fixes.**

Obstacle - It is important to prove to the end user their bug is fixed. It helps them.

Actions - I wrote a script that will help me quickly, simply, bundle the fixed code + their bad testcase + known good results + a README into a zip file. When they type ./RUNME.pl , the script will run the fixed code on their bad testcase and help them understand the new behavior of the fixed code.

Results - End users said they were delighted. Especially when the new behavior of the fixed code was not what they really wanted. This simple proving of bug fixes kept them informed and talking with me about what they really wanted.

Skills – Perl, code linting, customer relations, regression testing, automation.

Summary - Delighted customers by providing them testing collateral and simple scripts that proved fixes' behaviors.

Situation - **Our SW product was not being statically checked.**

Obstacle - Our SW product was a static code checker that itself was not being statically checked: the irony!

Actions - I learned an open source static check tool and tried integrating it into our flow.

Results - It did NOT work, our SW product is architected in a manner that prevents the open source static check tool from correctly running. It took 3 days to prototype and learn there was a bad fit issue.

Skills – Perl, prototyping.

Summary - Prototyped-and-Failed-Rapidly when trying to statically check the Perl code of the SW.

Situation - **Simulations needed a save feature, but senior engineering did not know how to accomplish that.**

Obstacle - No one knew how to do this, nor had the time to learn.

Actions - I championed this. I used an open-source tool that pauses a program and saves it. Our simulation architecture hindered both saving and reloading; I had to modify the open-source code to accommodate our strange architecture needs.

Results - We ran the simulation for 8 hours and then saved it when it hit the "reset" time. All engineers then began using that saved simulation if they tested something beyond "reset" time and they all saved 8 hours of runtime per engineer, per test.

Skills – C++, rewriting an open source project, RTL simulation, automation.

Summary - Added a save-feature to a CAD simulation SW.

Situation - **The team had stopped regression testing old versions of SW that were still in use.**

Obstacle - Users of old versions would issue bugs against changes the team was releasing. This was rework and also rejection of the team's fixes.

Actions - While ramping up on the team, I wrote a feature that bundled the regression tests against SW version numbers. The feature needed 2 parts. When running regressions, the correct 1-of-many versions must be executed. When writing tests, it must be easy to write many versions.

Results - My first significant task on the team was owning this new feature and proving to end users that our fixes met their needs. End users saw quality go up, so incoming bugs went down.

Skills – Perl, regression testing, automation.

Summary - Developed version matching between SW versions and the different versions of regression collateral.

Situation - **The team took on someone else's work in control registers.**

Obstacle - The team was understaffed already. The work was "be a middle layer" between end-users and the engine writers.

Actions - Two of us wrote a parsing language and a parser that took the end-users' domain specific language and generated Verilog code.

Results - The end-users were delighted to learn their task had been abstracted to a level they felt comfortable with.

Skills – Perl, C++, RTL coding, delegation, abstract SW, automation.

Summary - Developed an abstract parsing language that hid details from the users and made their job easier.

Personal Growth in Other Areas.

Situation - **Intel lacked an internal Makers club.**

Obstacle - Employees had no access to information concerning Intel's new IoT devices, no outlet for creative energies, and no out-of-the-box experiences.

Actions - I started a Makers club that met weekly and shared information on current trends in Making. This attracted support from Corporate Affairs who would invite the club to assist their volunteer opportunities. This also attracted the Intel IoT group who would invite the club to beta test their products.

Results - I created a club that met after work and enriched each other, the community, and official events. I learned some soft skills from running this club.

Skills – Leadership, mentoring, teaching.

Summary - Created and led a Maker's club that performed volunteer outreach.

Situation - **There's not enough heavy metal music in my life.**

Obstacle - Keeping 2 cars, 3 stereos, and a garage band PA system up-to-date with the latest music takes effort.

Actions - I set up a streaming media server that streams my own music to me, where ever I am at. I wrote SW for embedded computers in each car that connect to the car stereos and automatically play music. I created an Arduino shield that adds automation features to those embedded computers. And I set up the house server to support automatic code and music file updates to both cars when I park them at home.

Results - I don't sneaker-net my music anymore. If I am at work, I use my streaming media server to listen to my music. If I am in the car, I know last night the car's computer downloaded SW updates and music updates from the house server. What efforts do I now expend? I only continue the "buy new music and put it in the new music folder" task.

Skills – Perl, automation, IoT device development, Linux computing.

Summary - Automated my music hobbies.

Situation - **There's not enough beer and beef in my life.**

Obstacle - Making beer and smoking brisket is time consuming and detailed.

Actions - I built a beer machine. I use an Arduino, a laptop, custom SW and custom HW to automate beer making and meat smoking in my house. I can put ingredients into both devices, run 2 instances of the same SW, and press the GO button. And walk away; I get a text message when either is finished.

Results - I now have an automated process that turns $$$ into high quality beer and smokey beef.

Skills – Perl, automation, IoT device development, Linux computing.

Summary - Automated beer making and meat smoking hobbies.

Situation - **Writing SW is creative and hard.**

Obstacle - It takes serious effort to write quality SW.

Actions - I brought best practices home. I use dokuwiki for my personal notes, git for a code repository, bugzilla for tracking bugs I find in my hobbies. I also wrote a test framework that runs nightly regressions.

Results - I am enjoying the best practices at home as well as on the job.

Skills – dokuwiki, git, bugzilla, regression testing.

Summary - Uses industry best practices in his hobbies.

Situation - **An outreach team wanted to teach kids to program.**

Obstacle - No one on the team really knew how to do that.

Actions - I bought 4 identical laptops and setup them up with open source programming tools. I found some Arduino projects' documentation and code. As a team, we taught kids to program and have fun.

Results - I now have a kit that helps me teach 4 groups of kids how to program.

Skills – teaching, IoT device design.

Summary - Volunteers with STEM outreach programs and automates tasks there too.

Situation - **Intel employees wanted to build IoT devices. I wanted to learn to teach people technical things.**

Obstacle - There is a large learning curve for IoT. It takes practice to learn to teach people things.

Actions - I bought a wifi router and some IoT devices. I learned the basic problems and solutions of the IoT devices. I wrote a slide deck and collection of code. I began teaching at Intel an after-work class on IoT devices.

Results - Fellow employees asked me to help them on their jobs. Fellow employees stated they were encouraged to learn the IoT devices at home.

Skills – teaching, IoT device design.

Summary - Teaches fellow employees how to built IOT devices for their jobs and/or hobbies.

Setup and maintained a personal computer network at home.
Situation - During college, I had a dream of a large home network.

Actions - When I'd gotten a job, I saved up for a house. When I'd purchased a house, I saved up for a computer network. I learned the skills to keep the network running.

Results - I have several video gaming PCs, several servers, and a PC entertainment center. LAN parties are very easy to set up at my house.

Skills - Video gaming, networking, making dreams come true.

Summary - Built and maintains a fun video gaming network at home.

Done interesting things with Linux.
Situation - Linux is an interesting OS with good tools.

Actions - I created a script that incrementally backs up data between directories and computers. I learned Samba, SSH, and DHCP daemon usage.

Results - I recycled obsolete PCs into Linux file servers (with data protection and backup) for my family and friends. My father began to trust his PC enough to start telecommuting into work.

Skills - Encouraging people, Linux scripting, Linux networking.

Summary - Learned enough Linux skills to support computer networking needs.

Found a new hobby and embraced it whole-heartedly: Beer and sausage.
Situation - I wanted a new hobby and I chose the making of beer to be that new hobby. Later I chose to make sausage as well.

Actions - I bought the basic equipment and a beginner's handbook. I made several batches of beer and quickly learned and wrote down my mistakes and my version of the book's process. I bought books meant for intermediate brewers and I bought better equipment. I improved the documentation of my process. I got to a point at which I could examine my process; I wrote down my goals, needs, and resources. I created a new process that met my goals and needs with my given resources.

Results - I make good beer. I make good sausage.

Skills - Two fun new hobbies, basic chemistry, basic biology, process analysis, rudimentary process engineering, documentation.

Summary - Started two fun new hobbies and used engineering skills to become really good at them.

Making DJ work "Bigger, Brighter, Louder".
Situation - A friend was holding a convention and I volunteered to be his DJ and bring the PA. I asked him of his needs. I then acted upon a long standing dream I'd had: how to build a portable and near ideal PA?

Actions - I bought some professional grade equipment from Guitar Center. I bought some raw speakers from Fry's. I made 10 speaker boxes and put loading straps on each one. When I got to the convention center, I strapped 1 speaker box to every concrete pillar in the room. I hooked them up and setup the PA.

Results - The announcements were audible and no one complained of the sound system being too loud. And some people heard Weird Al Yankovic songs they'd not known existed.

Skills - Carpentry, planning, imagination, turning a dream into reality, PA work.

Summary - Designed a custom public address system to meet the needs of philanthropy adventure and for personal hobbyist needs.

Worked with Tek Marketing to create a phenomenal presentation for Intel's ISEF.

Situation - Tektronix wanted to set up a booth at Intel's International Science and Engineering Fair and the marketing division asked the engineers to help out.

Actions - I built up one quarter of the booth. I bought and cut open a Microsoft Xbox and soldered a logic analyzer probe set to the memory bus. I negotiated with Tek's Video Product Line to borrow a video signal analyzer. I soldered an oscilloscope probe set to the joysticks. Marketing allowed me to use a PA system and a plasma TV.

Results - On every day of the Science Fair, hundreds of high school students stood in front of the word "Tektronix" and played video games. I spoke to many of them about computer technologies. We all had fun.

Skills - Fine pitch soldering, entertainment, teaching,

Summary - Built an edutainment display meant for supporting Tektronix marketing in creating a phenomenal presentation for Intel's ISEF.

Modified a video game to make my nephews happy.
Situation – My nephews would complain when their video game characters were killed while playing on my server.

Actions - I copied someone's starship code and refactored it into a battleship and added it to the maps on my server.

Results – My nephews stopped crying over Skype about nerfing the bots on the server. And now they have the highest scores in the video game.

Skills – object oriented programming

Summary – Improve the skills of my nephews by modding the video game and adding a giant battleship.

Automated my online resume.
Situation – I have multiple copies of my resume in too many places and I only want 1 root document. And I want my resume online as well.

Actions - I added a resume page to my personal webserver. I once wrote a program that, given an Open Office document (my resume), will generate a PDF and move that PDF to a specified location. All I had to do was add the details of my resume and 3 target directories to the hash of the program to get it automatically converted each night. I also added my Leave-Behind Collateral files too.

Results – Through automation, I now update 1 copy of my resume and at midnight, it is converted into a PDF and copied to 3 key locations on my webserver, fileserver, and thumbdrive. I no longer have to remember where to place copies of the PDFs.

Skills – automation, Perl programming

Summary – Automated the conversion of my resume from *.doc into PDF so all release directories always contain my latest resume.

Wrote a "provisioning" SW.
Situation – I have many computers and use "disk ghosting" to keep 1 backup hard drive image. This means restoring 2 computers' hard drives causes them to have the same name and Windows Serial Number.

Actions - I had just learned C#. So I wrote a provisioning SW that has a few combo boxes and a "Setup" button. It also opens and saves an XML data file as well. It calls a MS Windows program that performs Serial Number registration. And finally, it calls a program that updates my desktop image with the name and details of the particular computer.

Results – Now, renaming a computer is much easier in my house. I practiced C# and hid 10 steps in a program with a nice GUI and data files.

Skills – automation, C# programming

Summary – Automated the renaming of computers by writing a provisioning SW in C# that handles all the steps necessary to rename and register a computer with Microsoft.