A Guide to Technical Resumes

Jack G. Ganssle
jack@ganssle.com

The Ganssle Group
PO Box 38346
Baltimore, MD 21231

How fast things change! Not long ago I got a dozen or more emails a day from companies and recruiters looking for developers. Some were so desperate for people they were willing to pay extraordinary salaries and bonuses. Wags were pronouncing massive shortages of engineers over the coming decade.

Now I'm flooded with mail from developers who have lost their jobs or feel their company is in danger of failure. Rare indeed is the message from a company trying to hire.

Macroeconomics mostly baffles me, but it does seem there's a pattern to our industry's boom and bust cycle. The early 70s, 80s, 90s and now 00s all saw recessions of greater or less magnitude. For some reason the start of a new decade brings a slowdown. Those of the 80s and 90s even changed the political climate as incumbent presidents lost their reelection bids.

Electronics always gets hit hard in these recessions. No surprise there; there's little special about our business and so it's natural it too should decline in sync with the rest of the economy. The 90s subordinated common sense to a fury of stock market irrationalities. Many analysts sagely pronounced high-tech as being recession-proof. Too many folks believed them and spent as fast or faster than they earned.

Lifetime employment, once a staple of businesses like IBM, is long gone. There is no job security anywhere anymore. Despite Dilbert's buffoon-like portrayal of managers, I sure don't envy their roles. Can you imagine responding to non-negotiable forces like demanding stockholders, or empty bank accounts? By far the biggest expense for most high-tech companies is salaries. So there's little surprise that's the first place most outfits economize. It's awful and brutal, and horribly impacts people's lives, but in our capitalistic economy I can conceive of no alternative. Some would say the problem stems from the greedy Enron culture where the top dogs earn (or steal) millions or even hundreds of millions. Perhaps. Surely that's worth cleaning up. But a problem that's a few tens of billions in the United States' $10 trillion economy is in the noise.

The moral is that bad times always come. A hot economy is the prelude to a downturn. To us little people that means being prepared for the inevitable troubles. Squirrel away money. Avoid credit card debt. Bucks in the bank give you options and flexibility when the company folds, or when all engineers are told to take a 10% pay cut "for now".

Lawyers tell us to audit insurance policies and wills regularly. Nothing wrong with that, but I think it's more important to manage life than death. In the best of times and in the worst of times routinely update and tune your resume. Pretend you are a tired boss tasked with hiring, made a bit cynical by digging through a pile of resumes with their exaggerated claims. How can you appeal to that person?

It's a competition, in a sense a battle against your peers. The resume is your main and perhaps only tool to get a foot in the door. It's the key that may get you admittance to the interview. Post-interview, most employers review the resume, make notes on it, highlight the good stuff and problem areas. It's circulated for comments.

A crummy resume – and in bad times, anything that's less than stellar – will doom your job search.

Sales and Marketing

This critical document is a selling tool. A lot of us hate the thought of sales and marketing. I remember as a very young and very naïve engineer telling a group of older folks how engineering is so "pure" and unsullied by the grittiness of sales and marketing. They all laughed, and rightly so.

Everything we do is sales. How do you convince your boss to get new development tools? Sell him. Show how the benefits outweigh the costs. Want to get your colleagues to start using UML or eXtreme Programming? Better sell them, hard, since change is always difficult. Show them the upside of the change. Be prepared to push for some time, as the biggest changes need the most selling. Hit them on all fronts.

Successful sales means we must speak the customer's lingo. Too many engineers never get this, and talk to their bosses about bits and bytes when those individuals really want cost/benefit ratios. Write your resume to communicate clearly what you've done to someone who probably doesn't have a clue about your specific field.

Acronym overuse is a mistake. None of us know them all; worse, a lot are industry-specific. Few folks not building colorimeters, for example, know what CIE means. It's best to describe your projects in terms any working engineer knows.

An example snore-inducer: "Worked on DOB-EKV project, used Shear/Mellor in C++ on Galazor 3.12 compiler running CDC1412."

Yuk. Who cares what compiler, let alone version, you used? What did the system do? Who used it? Did it work? Was this a big job… or did it take you 6 months because you're incompetent?

Better: "Wrote the DOB-EKV star tracking software for Marshall Space Flight Center. Used Shear/Mellor methodology and an object-oriented design. I wrote 25k lines of C++ running on an ARM processor. Six month project that flew successfully."

Most resumes start with career goals. They're a waste of space, and are better handled in the cover letter. Let's face it: the prospective employer, at this early stage, cares little about you as a person. Most are thinking only in terms of "can he do the work?"

Instead, start the resume with a well-written, hard-hitting career summary that tells the reader (in one sentence) what sort of job you're looking for. Follow that with a sentence or two that states what you're really good at, that tells the company how they can use you effectively: "Though I have written over 50k lines of GUI code in the Windows environment, I'm one of the best 8051 assembly and C programmers around. I can build tight, fast interrupt handlers for you, and am a master of working with an RTOS. If your product has performance constraints, limited memory resources, or tight timing, I'm the man for you."

Traditionally this sort of salesy prose goes into the cover letter. That's a mistake. They get only a fraction of the scrutiny devoted to the resume. Figure anything in the cover letter will be forgotten or de-emphasized, so make sure all of the meat, everything you want to say, is in the resume itself.

The second paragraph should be a summary of your skills, written in clear concise English with a minimum of jargon. It's never a laundry list of languages and tools we've used; rather, write an honest (and salesy) assessment of what you're good at. Don't pretend to be an expert at everything. No one is. Consider this format: "Expert at C (have written over 200k lines), C++ (150k), assembly (100k). Very good at Perl, Fortran. Have completed projects in Modula, Pascal, Ada, and Algol." The evaluator will appreciate the easily read information and the honesty.

In the experience section, list your jobs chronologically, working from the present to the past. Always note the start and end dates, including the month, for each job. Any attempt to paper over an employment gap will be discovered so don't even try to fudge the dates. And please do include your job title.

Describe what you did – without acronyms and jargon – at each job. Pretend you're reading the resume; what would you like to see in a candidate? I want to see what the candidate actually did. That means write with active voice. Tell a (short) story about your accomplishments, citing examples that scale to other companies.

Usually the resume ends with the usual long, dreary and boring list of languages, environments and tools you've used. Skip this. No one cares. If you've followed my advice the reader already knows the highlights of this. A wise reader realizes that anyone can learn how to use an ICE or a new IDE.

If you have other affiliations or experience that is *relevant to a prospective employer*, by all means include the information. Be specific. Add dates, dollars, or other quantitative information.

Resumes targeted at US companies should be 2 or three pages long. A single page is appropriate for a new grad; 5 pages is too much. Note, though, that overseas longer is better; 5, 7 or even 10 pages may be OK if you have a lot of experience. At that length figure on more descriptive prose and fewer bullets.

Include lots of contact information. The experienced resume reader expects a pretty high BS factor. Prove your points by including references (don't make the reader ask for them), and phone numbers and contact names for every job. The ideal reference is a former supervisor. Always include the reference's relationship to you, thus cheating the reader's natural assumption that these are your best buddies. Find powerful, compelling references. Don't have any? It's your responsibility during your career to develop these relationships. Find and nurture people who can, over the years, aid you in job searches.

If you had one bad experience, leave the contact information off for that one job and be ready to explain the circumstances, honestly, in the interview.

Everyone understands that if you're looking around while still employed you'd rather not have the current boss contacted, but it's best to be explicit about your wishes.

Be sure it's easy to contact you. I've seen resumes with neither email nor phone. That hardly speaks well for someone wanting a cutting-edge technology job.

In the USA, if applying for an engineering/software job, don't call the silly thing a CV (curriculum vitae). That's pretentious, used mostly by people applying for academic jobs.

Don't include personal information about your kids, your hobbies or your pets. Believe me – no one cares. The space eaten up by these words is simply wasted. And worse, your avocation may be the resume-reader's pet peeve.

Honesty

Never, ever, lie on the resume. The truth has a nasty habit of surfacing. These days companies check on degrees, immigration status, and more. For a few hundred dollars you can now have anyone investigated in at least surface detail. Many companies routinely run such checks.

But an honest approach doesn't mean you can't practice a bit of truth management. For example, why not customize the resume for a particular job opening? If you know the firm uses 68HC11s, and you have that experience, emphasize it. Devote a greater part of the document to your expertise with this processor. If you have no 68HC11 background stress the transferable skills: C is pretty much the same thing on any small microcontroller. Sell the reader on your competence in their area of interest.

You can go too far with the truth. One resume I've kept for years was for a person applying for a medical job. Under "mental health" she wrote "better, now that I have full custody of my kids", and then goes on for several paragraphs about the details of her breakdown. Discretion is the better part of valor.

I worked for years with an engineer who later admitted he had no analog experience, though was hired as an analog designer. He told me: "I just lied about it all." Sure, he got the job, but that's a lousy approach to life. And he was quite awful at the work, later being relegated to an almost clerical position.

Editing

I've read hundreds, maybe thousands of engineers' resumes over the years, and am struck by how many are so poorly edited. If you can't be bothered to get the tedious language details right, then who's likely to believe you'll take the time to make perfect software?

Write it and then proofread it. Put the doc on a shelf and reread it a week later. You'll be amazed at the mistakes that leap off the page. Have several friends go over it; if one's an English major, so much the better. Make sure a techie pal or two checks the details as well.

Try to get a boss-type, one who actually hires people, to edit the resume. That input can be invaluable. Remember your target audience isn't yourself or technical peers – it's someone who's making a selection based on how well the document matches their needs.

The linqua franca in the USA is English, yet so many of these important selling documents are written in what appears to be a corrupt technical argot. Butcher the language and you'll surely convince readers of your poor education. Which seems a very bad way to apply for a job.

There are important differences between your and you're, between there, their and they're. A downtown billboard shouts the ad agency's thoughtless approach to work due to the mix-up of your and you're. These sorts of errors are inexcusable and easily avoided, and are deadly on a resume.

It's OK to mangle the grammar a bit; clarity and brevity is much more important than a gripping storyline. Don't be chatty but do use active voice. Convey the information in an easy-to-read concise way.

We all want our resume to stand out, but resist the urge to be too quirky. Don't print it on bright orange paper. Don't include a photo of your dog or a bio of your spouse. I've seen all of these. It's a sure way to put off the suits who often make the big decisions.

When times are hard want ads generate a resume avalanche. Stand out by carefully matching your skills to the prospective employer's needs, and by concisely, cogently and honestly documenting your prowess.

Joe Coder

1 Coder Court, Coderville MD 21231
(410) 555-3647 (home)
(410) 555-1214 (work)
joe@hotmailyahoo.com

Career Summary

My dream job is being part of a small team building great 8 or 16 bit embedded systems in C and assembly.

Though I have written over 50k lines of GUI code in the Windows environment, I'm one of the best 8051 assembly and C programmers around. I can build tight, fast interrupt handlers for you, and am a master of working with an RTOS. If your product has performance constraints, limited memory resources, or tight timing, I'm the man for you.

I'm an expert at C (have written over 200k lines), C++ (150k), assembly (100k). Very good at Perl, Fortran. Have completed projects in Modula, Pascal, Ada, and Algol. Greatly experienced with the VxWorks and uC/OS real time operating systems, as well as with the CAN bus and internet protocols. I've used UML on big projects and am a certified UML trainer.

Experience

The Code Dudes – Chief Programmer (4/00 to Present). Supervisor Jack Dude, 410-555-3322.
I designed, coded and tested all of the firmware in the company's Wacko 2000 ground beef analyzer, which used 12 68HC11 processors communicating over a CAN bus. 35K lines of C developed in 6 months using Lint and code inspections. Delivered 2 months ahead of schedule and 8% under budget. Handled picoamp-level analog inputs at microsecond rates; developed unique FIR filters to smooth the noisy raw data. I became an expert using uC/OS in this 14 task system. I implemented an embedded web server for remote maintenance of the product using a commercial TCP/IP stack.

The Code Dudettes - Programmer (1/97 to 4/00). Supervisor Jacky Dudette, 410-555-5435.
I was part of a three person team that designed, coded and tested the firmware in the company's range of colorimeter products. I implemented firmware on the 32 bit PowerPC, the 16 bit 186, and the 8 bit 68HC05 and Z180. I wrote some 150K lines of flash-based C++ on the PowerPC product, and over 100K lines of C on the 8 and 16 bit processors.

The PowerPC product used VxWorks and was developed using Wind River's Tornado environment. We modeled the entire system in UML before generating code, using Rhapsody to back-annotate changes into the model.

The Z180 system was resource and timing constrained. I wrote it entirely in 32k of assembly language as it had to respond to events at 50 usec intervals. The system simulated a phase-locked-loop, so real-time response and analog stability were critical issues.

Other Affiliations

Member, Advisory Board, Dufuss Corporation, 1997 to present. In this role I help the startup evaluate technical options such as language selection, tool use, and advise their programmers on appropriate process issues.

Publications and Papers

I presented three papers about UML modeling at the 2002 and 2003 Embedded Systems Conferences. My article "Extreme UML" ran in Embedded Systems Programming in August, 2002.

Education

- BS in Computer Science, 1996, University of Southern North Dakota. Top third of class
- Four weeks of UML training from iLogix, 2000
- Three weeks of training from Wind River on the Tornado environment and VxWorks.

References

- Jack Dude, former supervisor. 410-555-3322
- Jacky Dudette, former supervisor 410-555-5435
- Professor John Seagul, head of CS department at my alma mater, 410-555-6587

Better Firmware... *Faster!*

For Engineers and Programmers

This seminar will teach you new ways to build <u>higher quality</u> products in <u>half the time</u>.

80% of all embedded systems are delivered late…

Sure, you can put in more hours. Be a hero. But *working harder is not a sustainable way to meet schedules.* We'll show you how to plug productivity leaks. How to manage creeping featurism. And ways to balance the conflicting forces of schedules, quality and functionality.

… yet it's not hard to double development productivity

Firmware is the most expensive thing in the universe, yet we do little to control its costs. Most teams deliver late, take the heat for missing the deadline, and start the next project having learned nothing from the last. Strangely, *experience* is not correlated with *fast*. But *knowledge* is, and we'll give you the information you need to build code more efficiently, gleaned from hundreds of embedded projects around the world.

Bugs are the #1 cause of late projects…

New code generally has *50 to 100 bugs* per thousand lines. Traditional debugging is the *slowest* way to find bugs. We'll teach you better techniques <u>proven</u> to be up to 20 times more efficient. And show simple tools that find the nightmarish real-time problems unique to embedded systems.

… followed by poor scheduling

Though capricious schedules assigned without regard for the workload are common, even developers who make an honest effort usually fail. We'll show you how to decompose a product into schedulable units, and how to use killer techniques like Wideband Delphi to create more accurate estimates.

Learn from the Industry's Guru

Spend a day with Jack Ganssle, well-known author of the most popular books on embedded systems, technical editor and columnist for Embedded Systems Programming, and designer of over 100 embedded products. You'll learn new ways to produce projects *fast* without sacrificing quality. This seminar is the only <u>non-vendor</u> training event that shows you *practical* solutions that you can implement *immediately*. We'll cover technical issues – like how to write embedded drivers and isolate performance problems – as well as practical process ideas, including how to manage your people and projects. *After taking this class you'll receive a certificate awarding you 0.7 Continuing Education Units.*

Seminar Leader

Jack Ganssle has written over 600 articles in Embedded Systems Programming, EDN, and other magazines. His five books, **The Art of Programming Embedded Systems**, **The Art of Developing Embedded Systems**, The **Embedded Systems Dictionary**, **The Firmware Handbook**, and **Embedded Systems, World Class Designs** are the industry's standard reference works

Jack lectures internationally at conferences and to businesses, and has been the keynote speaker at the Embedded Systems Conferences in both Boston and San Francisco. He founded three companies, including one of the largest embedded tool providers. His extensive product development experience forged his unique approach to building better firmware faster.

Jack has helped over 600 companies and thousands of developers improve their firmware and consistently deliver better products on-time and on-budget.

Course Outline

Languages
- C, C++ or Java?
- Code reuse—a myth? How can you benefit?
- Controlling stacks and heaps.

Structuring Embedded Systems
- *Manage* features… or miss the schedule!
- Using multiple CPUs.
- Five design schemes for faster development.

Overcoming Deadline Madness
- Negotiate realistic deadlines… or deliver late.
- Scheduling - the science versus the art.
- Overcoming the biggest productivity busters.

Stamp Out Bugs!
- Unhappy truths of ICEs, BDMs, and debuggers.
- *Managing* bugs to get good code fast.
- *Quick* code inspections that keep the schedule on-track.
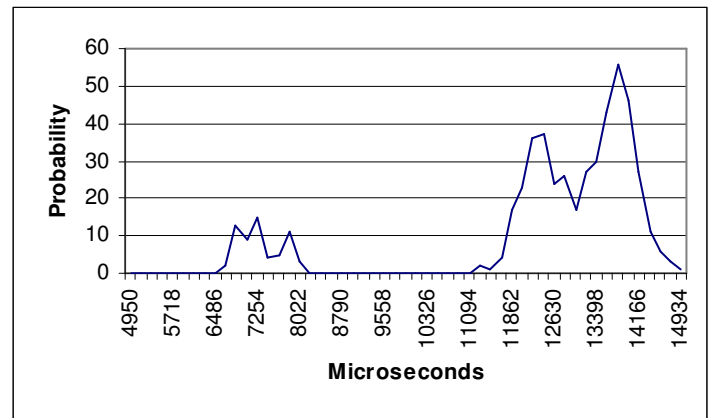- Cool ways to find hardware/software glitches.

Managing Real-Time Code
- Design *predictable* real-time code.
- Managing reentrancy
- Troubleshooting and eliminating *erratic crashes*.
- Build better interrupt handlers.

Interfacing to Hardware
- Understanding high-speed signal problems.
- Building peripheral drivers faster.
- Inexpensive performance analyzers

How to Learn from Failures… and Successes
- Embedded disasters, and *what we must learn*.
- Using postmortems to accelerate the product delivery.
- Seven step plan to firmware success.

Do your routines execute in a usec or a week? This function is all over the map, from 6 to 15 msec. You'll learn to write real-time code proactively, finding timing issues early.

Why Take This Course?

Frustrated with schedule slippages? Bugs driving you batty? Product quality sub-par? **Can you afford *not* to take this class?**

We'll teach you how to get your products to market faster with fewer defects. Our recommendations are *practical*, *useful today*, and *tightly focused* on embedded system development. Don't expect to hear another clever but ultimately discarded software methodology. You'll also take home a 150-page handbook with algorithms, ideas and solutions to common embedded problems.

Here is what some of our attendees have said:

Thanks for a great seminar. We really enjoyed it! We're already putting the ideas you gave us to use.
J. Sargent, CSC

I like your practical, no nonsense advice backed up with numbers, your dynamic presentation style, and the nice handout that you gave us. I will definitely recommend your seminar to other programmers.
Ed Chehovin, US Navy

I just wanted to say thanks for a great seminar last week. Already the information you gave has proven useful – I used that ISR trick and we finally found an error we've been chasing for months.
Sandeep Miran

Thank you so much for a great class! Now my co-workers think I'm the guru!
Dana Woodring, Northrup Grumman

Did you know that…

… doubling the size of the code results in <u>much more than twice the work</u>? In this seminar you'll learn ways unique to embedded systems to partition your firmware to keep schedules from skyrocketing out of control.

… you can <u>reduce bugs</u> by an order of magnitude <u>before</u> starting debugging? Most firmware starts off with a 5-10% error rate – 500 or more bugs in a little 10k LOC program. Imagine the impact finding all those has on the schedule! Learn simple solutions that don't require revolutionizing the engineering department.

… you can create a <u>predictable</u> real-time design? This class will show you how to measure the system's performance, manage reentrancy, and implement ISRs with the least amount of pain. You'll even study real timing data for common C constructs on various CPUs.

… a 20% reduction in processor loading slashes development time? Learn to keep loading low while simplifying overall system design.

… few watchdog timers are properly implemented? Most are partial solutions to a complex problem. We'll show you how to build an awesome WDT.

… most interrupt-driven timers are improperly coded? Subtle asynchronous issues always lead to erratic timer reads and crashes. The solutions are not obvious, but easy to implement.

… reuse is usually a waste of time? Most companies fail miserably at it. Though promoted as the solution to the software crisis, it's much tougher than advertised. You'll learn the ingredients of successful reuse.

Busy Schedule? . . .

If you can't take the time to travel, we can present this seminar at your facility.

We will train all of your developers and focus on the challenges unique to your products and team.

Thanks for a valuable, pragmatic, and informative lesson in embedded systems design. All the attendees thought it was well worth their time.

Craig DeFilippo, Pitney Bowes

Thanks for the terrific seminar here at ALSTROM yesterday! It got rave reviews from a pretty tough crowd.

Cheryl Saks, ALSTROM

I just wanted to thank you again for the great class last week. With no exceptions, all of the feedback from the participants was extremely positive. We look forward to incorporating many of the suggestions and observations into making our work here more efficient and higher quality.

Carol Batman, INDesign LLC

Contact us for info on how we can bring this seminar to your company

E-mail: info@ganssle.com
or call us at 410-504-6660

What are you doing to upgrade your skills? What are you doing to help your engineers succeed?

Do you consistently produce quality firmware on schedule? *If not . . . what are you doing about it?*

Better Firmware... *Faster!*

A one-day class
April 23, 2008 - Chicago
April 25, 2008 - Denver
May 19, 2008 - London

Spend a day with Jack Ganssle, Embedded System Programming's Technical Editor and columnist, and learn new ways to <u>get your products to market faster</u>, and improve your resume with the 0.7 Continuing Education Unite you'll be awarded.

Registration Information

All of this, plus 150 pages of handouts, for just $695 per person. Plus you will receive a personalized certificate of completion at the end of the course.

Groups of 3 or more registering together pay only $595 each.

Register early and save. Sign up a month in advance, and receive a $50.00 discount.

Fax this form to 647-439-1454. Or, register by phone at 410-504-6660 or via email to register@ganssle.com.

Cancellations made more than 14 days prior to the class are refundable less a $50 fee. Cancellations made within 14 days are non-refundable, but are 100% transferable to all courses we offer.

April 23, 2008
Chicago, IL
Hilton Oak Lawn
9333 Cicero Ave.
Oak Lawn, Chicago

April 25, 2008
Denver, CO
Embassy Suites Hotel
7525 East Hampden Ave.
Denver, CO

May 19, 2008
London, UK
Jurys Inn Heathrow
Eastern Perimeter Rd
Hatton Cross, Hounslow

Registration Form

Today's Date: _____

Name: _____

Company: _____

Mailing address: _____

City, State, Zip: _____

Phone: _____   Extension:_____ ___

Fax: _____

Email: _____

Location: Chicago _____Denver _____London _____

Number of attendees: _____

Purchase Order Attached. P.O. Number: _____

Charge to: Visa   MasterCard   American Express

Card Number: _____ Expires: _____

Name on Card: _____

Signature: _____

Fax this to 647-439-1454. Or, call us at 410-504-6660.