Extracted from:

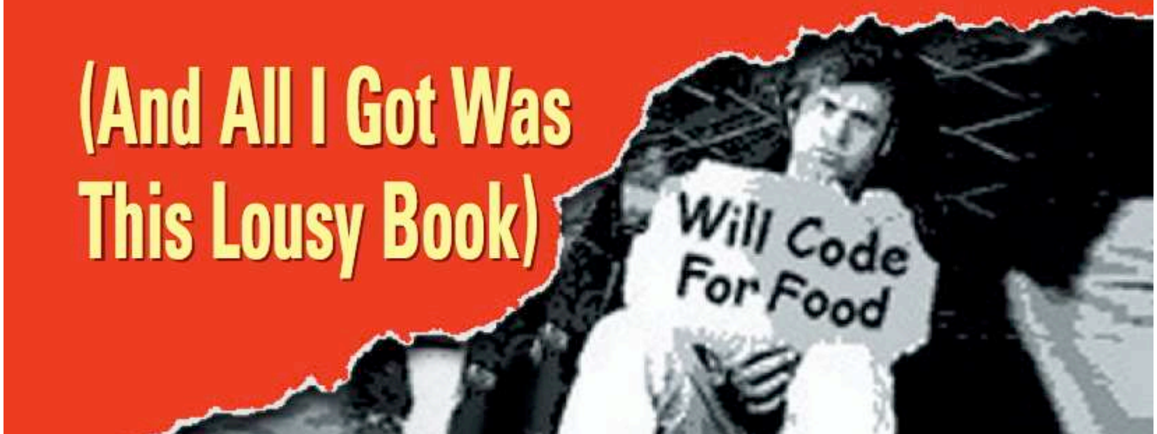
My Job Went to India

And All I Got Was This Lousy Book

The
Pragmatic
Programmers

MY JOB WENT TO INDIA

(And All I Got Was This Lousy Book)



52 Ways To Save Your Job

Chad Fowler

It is true India has the advantage in software and China in hardware. If India and China cooperate in the IT industry, we will be able to lead the world...and it will signify the coming of the Asian century of the IT industry.

► Wen Jiabao, Chinese Premier, April 11, 2005

Introduction

I awoke to an odd smell. *Where am I?* I asked (aloud, I think).

I was in Bangalore, India's Garden City. That odd smell was the remarkably foreign combination of pollution, ultraspicy food from the hotel's kitchen, and something else that I could never quite put my finger on. It was my first morning there, and I was late for work. I didn't feel bad about that, considering the hellish 32-hour journey I had suffered to get there. And that my India-savvy co-worker tricked me into an all-night cultural immersion in the world's scariest hotel, after getting off the plane in Bombay the night before.

After coming to, I felt a panic, realizing that my driver must have been waiting downstairs for an *hour*. *Oh God, he'll be angry*, I thought, as I scrambled to get ready for the first in a series of all-day interviewing sessions. *That's just what I need on my first morning in a place like this...an angry taxi driver.*

I rushed downstairs, resisting the fabulous aroma of a South Indian breakfast, and ran out to ask the doorman to page my driver. I asked if the driver had been there long. He had. Two hours. *Ugh*.

I spent the first five minutes in the car with Joseph apologizing profusely over being late and making him wait. He laughed dismissively. *This is my job. I wait all day.* And as I found out later, he really did. He didn't just drop me off at work and come back at a fixed time. He waited at the office until the very minute I was ready to go. Without warning, at any time, I could come down from the office and expect to hop into the car and be driven away.

My first exposure to India in daylight was that drive across town from the northwestern corner of Bangalore to the southwestern corner. The culture shock started to hit me.

Bangalore is known as India's Silicon Valley. Being from a small city back home, it was exciting to realize that I had come to a technical mecca.

More surprising, though, were the extreme contrasts between high tech and low tech. I saw half-naked children playing in the dirt in front of a huge Yahoo! sign. I saw a rickshaw with a Novell advertisement on the back and another with what looked like a Sun Solaris CD dangling from the rearview mirror as an ornament.

We drove past beautiful, state-of-the-art office buildings, filled with the employees of some of the Western world's most innovative companies. We dodged buffalo in the street and begrudingly yielded to rickety bicycles and full families on single mopeds.

We passed by fields containing huts made from twigs, mud, tarps, and assorted garbage. We drove through crowds of well-dressed young people, drinking coffee outside their office buildings before the start of the day, only to drive a little farther to be propositioned by lepers begging at a traffic light.

So before I even reached the office on my first day, my perspective had changed. This was a world of great extremes. These foreign voices I had heard through scratchy, unreliable phone connections, attached to the brains whose code I'd been ruthlessly reviewing, lived *here*? *These* are the people who are allegedly stealing our jobs?

I had come to India in the first stage of the setting up of a new software development center for my company. My job was to interview and select about 25 people who would form the "seed team" of a development shop that would eventually house 250 people. More precisely, my job was to *reject* more than 200 people. We had advertised our open jobs and received nearly 30,000 applications. That's four zeroes. You are reading it correctly. We hired outside firms to help us whittle the 30,000 down to a more manageable number and then used our own U.S.-based employees to further work that number down to a short list of a couple hundred that we could interview in person.

I was to be our interview panel's executioner, sniffing out the weak and finishing them off quickly and (I hoped) painlessly. While in India, I visited the hotel conference rooms of three different cities and met hundreds of people. I probably took a secret pleasure from the fact that I was going to go over and *stop* all these people from getting through the system and "stealing" our jobs.

It was post-boom time. By that, I mean the DotCom bubble had burst. The IT sector's lifestyle had gone from rock 'n' roll to Holiday Inn lounge act, and it was showing in India as well.

In fact, what I found was not an army of people, plotting to steal our comforts for themselves. Unlike their counterparts in the West, these people weren't angry that they had to get a small television set or even that they might not be able to afford this month's cable TV bill. These were sons and daughters who were scraping by, trying to raise money to support their parents and their spouses' parents. These were mothers and fathers whose IT jobs meant the difference between *really* educating their children or sending them to a school from which the further educational options have a *hard limit*. They weren't trying to steal the American dream. They were trying to squeeze a once-dry economy for a few drops of life-giving cash flow.

Ultimately, I was an executioner very much fit for the task. No physical injuries resulted, but many interviewees left with bruised egos. What I left with was a changed perspective. Things had changed. A vibrant society of *highly motivated* and intelligent people existed here. And they weren't playing for amenities; they were competing for the survival of their families.

You can't underestimate—or blame—someone with that kind of motivation.

Things Ain't What They Used to Be

According to the U.S. government, IT unemployment has doubled since 2000. The Bureau of Labor Statistics reports that between 2000 and 2004, the number of programmers in the American IT industry dropped by 17%. In just the first three months of 2005, U.S. techology companies cut 60,000 jobs—twice the number cut in the same period of the previous year.[1] The numbers are sobering. In this world in which every device seems to contain a computer, could software development be a doomed profession?

Matters are made more confusing by the bipolar temperament of the IT job market. Had you left and went on retreat in a cave in 2000 for several months, you would have emerged into an IT employment landscape that was as unrecognizable as Java to a COBOL programmer. In the mid- to late-nineties, a gold rush took place in the IT industry. I remember reading about employers giving BMWs as signing bonuses. A team from another company actually auctioned itself off on eBay for a huge signing bonus.

[1] "Challenger Tech Sector Job Cuts Report," http://challengergray.com/

IT employment was a seller's market, and people were jumping ship from other vocations in droves.

The market was suddenly being flooded with new talent (or, at least, people who considered themselves talented). At the time, demand was still outpacing supply. I saw Java programmers being shipped in from India who seemed to have passed the time during their flights reading their first Java manual. The flood surely gave passage to some great software people. But it also introduced a large population of people who wouldn't have ever considered a career in software—and probably shouldn't have.

We're all painfully aware that the boom has ended. When the bubble burst, it was as if an unruly bunch of children had been interrupted jumping on the bed and suddenly realized how much of a mess they had made of their room. It was a mess they now had to clean up. Our industry was filled with piles of unneeded software, hollow business models, and increasingly irrelevant people.

The turn of the century saw IT being demoted from knight to squire. Organizationally, CIOs had bubbled to the top, often reporting to their companies' CEOs. They were now being reorg'd back down under the COOs and CFOs where they started. And with this demotion came the budget cuts.

Where IT departments were previously under pressure to scoop up the best talent before the competition did, they were now under pressure to shed the excess baggage they had collected. In many cases, the reduction in force *didn't* come with a reduction in workload. Bubble or not, the technology boom made our businesses more reliant on IT than ever before. Business processes from the sales floor to the call center were now resting on the backs of IT's systems.

So, here we were with way too much work to do and way too few jobs to support all the work. What's a poor CIO to do?

"Offshoring." This silly-sounding made-up word now strikes fear into the hearts of IT professionals throughout the Western world. Too much work mixed with budget reductions leaves little choice for the nation's CIOs. Programmers in India can be hired for as low as a tenth of the salary of a programmer in the United States. And without a standardized, objective way to compare and contrast the talent difference, that's a bargain difficult for a smart business person to turn down. Even with the time zone and cultural differences, it's hard for a finance manager to imagine *not* saving real money with the right offshoring setup.

So, jobs have been shipped overseas by the boatload. Many American programmers have found themselves either unemployed or supporting the skeleton crew as one of the last of a dying breed. Early-morning or late-night teleconferences through fuzzy telephone connections with people who "talk funny" are becoming a common occurrence in the software development world.

And it looks like the burst of the bubble didn't make a temporary trough. This is the new IT landscape. Over the years since the boom, offshoring has been growing at a steady rate. In 2004, IT outsourcing grew by 37%.[2] And according to Gartner, a research and advisory firm, worldwide offshore spending on application development will more than double, reaching $50 billion dollars by 2010.

It's not just grunt work that's going, either. While we're already spending $1.2 billion on R&D outsourcing, that number is expected to shoot up to $12 billion by 2010.[3]

The IT offshoring boom has been historically associated with India. India started with a marked advantage over many other low-cost countries, largely because of its excellent educational institutions and, more important, the prominence of English as a first or second language. But even for India, competition is heating up. More and more business is being shipped to Eastern Europe (where it's easier to find multilingual employees to support European language–speaking nations), Russia, Malaysia, and the Phillipines, to name just a few.

Most recently, China has begun to figure into the equation. You know, China. They're the ones who manufactured almost everything in your house. Go to Wal-Mart, and try to buy a clock or a phone that wasn't made in China. It's a real challenge. And now, they've got some forward thinking Indians wondering how long the "offshoring bubble" has left in India. Leading management consultancy McKinsey & Company reports that although it will be some time before China could eclipse India in IT offshoring, progress is being made. Chinese offshoring revenues are increasing by 42% each year on average, and the number of English-speaking college graduates in the Chinese workforce has more than doubled since 2000.[4]

[2] http://management.silicon.com/itdirector/0,39024673,39127146,00.htm
[3] http://informationweek.com/story/showArticle.jhtml?articleID=160400498
[4] http://www.mckinseyquarterly.com/article_page.aspx?ar=1556&L2=4&L3=115&srid=21&gp=1

The bottom line is that things have changed for us professional software developers here in the Western world. All signs indicate that the change is not temporary. We can expect our DotCom bubble glory days to become a more and more distant memory as the world continues to turn to lower-cost sources of software development labor.

It's All Our Fault

It's easy to demonize Big Money America or criticize the government for not protecting us. Or, for the truly adventurous of imagination, it's easy to believe that Indians have developed some sinister plot to maliciously rob us of our comforts. However, even if there is an ounce of truth in these sentiments, it is outweighed by the pounds of mediocrity under which our Western industry has languished for the last several years.

It's understandable that forlorn programmers would dump their personal tragedies at the feet of anonymous companies and governing bodies. It's somehow comforting to drown one's fear or despair in a healthy helping of anger and strategically directed blame. And to make matters worse, media sensationalists such as Lou Dobbs prey on our fears, hyping up the problem and sounding a rallying cry whose primary purpose is to get better ratings. But ultimately, blaming corporations is a dead-end road. We can't change corporate America. And though we have democracy on our side, none of us can single-handedly steer this massive ship of a country.

So though comforting in times of fear and uncertainty, this game of blame-the-big-guy is fruitless. We have no one to blame but ourselves.

This self-blaming attitude isn't defeatist, though. In fact, blaming the government is the defeatist choice here. Forming labor unions and picketing would be defeatist. Sitting on the couch flipping news channels and cursing in a fit of nationalist rage would be defeatist. All these courses of action lay the blame—and the imperative for action—at someone else's feet.

If we can calm down enough to look at the situation rationally, we see that it is our own fault that we're in this mess. We live and work in an economic ecosystem. In ecosystems, it's the strong who survive. During a period of staggering success, we've allowed ourselves to get fat, lazy, and slow. The state of our craft has been marred by years of mediocrity.

The fact that we can take responsibility for and see the path that led to our predicament is a good thing. It means we can start to take (and own) corrective action.

It's Up to Us

It's time to face the music. We are where we are, and waiting for things to change by themselves isn't going to lead us anywhere different. The trends aren't reversing, and the government has no incentive to bail us out. The good news is that each of us has the power to do something about it individually. We can each take control of our own piece of the situation, bringing sanity to the collective whole.

Of course, if you can't stand the heat, the most obvious action is to get out of the proverbial kitchen. Western IT has its share of post-boom deadweight still lingering around, nervously drawing a paycheck. For some not-insignificant percentage of IT workers, the safest bet is to start looking for an alternate line of work. Choosing when you leave and where you go next is a lot less difficult than being thrown out. If you don't have passion and a drive that would force you to create software whether you were being paid for it or not, you're not going to be able to continue to compete with those who do.

For those who remain, here is the key to survival: Software is a business. We're going to have to be businesspeople. Our companies don't employ us because they love us. They never have, and they never will. That's not the job of a business. Businesses don't exist so we can have a place to go every day. The purpose of a business is to make money. To stay employed, you're going to have to understand how you fit into the business's plan to make money.

As we'll explore later, keeping you employed costs your company a significant amount of money. Your company is *investing* in you. Your challenge is to become an obviously good investment. If the *business value* you bring is clear, you are far less likely to end up on the offshoring chopping block.

Think of your career as if it is the life cycle of a product that you are creating. That product is made up of you and your skills. In this book, we'll look at four facets that a business must focus on when designing, manufacturing, and selling a product. And we'll see how these four facets can be applied to our careers:

1. **Choose your market.** Pick the technologies and business domains you focus on consciously and deliberately. How do you balance risk and reward? How do supply and demand factor into the decision?

2. **Invest in your product.** Your knowledge and skills are the cornerstone of your product. Properly investing in them is a critical part

of making yourself marketable. Simply knowing how to program in Visual Basic isn't good enough anymore. What other skills might you need in the new economy? How can you compete with both your offshore and onshore rivals?

3. **Execute.** Simply having employees with a strong set of skills doesn't pay off for a company. The employees have to *deliver*. How do you keep up the delivery pace without driving yourself into the dirt? How do you know you're delivering the *right* value for the company?

4. **Market!** The best product in history won't get purchased if nobody knows it exists. How do you get find recognition in both your company and the industry as a whole without "sucking up"?

The goal of this book is to give you a systematic way of approaching the challenges that lie ahead of you in the new world of IT. We will walk through specific examples and present a set of actions that you can take *right now* that will have both short-term and long-term positive effects.

Ultimately, the goal is not to bring our jobs back. These low-value jobs we've lost were *meant* to be sent offshore. Instead, we should be preparing for the new wave of higher value jobs that will be created in their places.

Acknowledgments

I would have never written a book if not for Dave Thomas and Andy Hunt. *The Pragmatic Programmer* [HT00] served as a catalyst for me, and I've been inspired by their work ever since. Without Dave's encouragement and guidance, I would have never believed I was qualified to write this.

Juliet Thomas served as an editor early in the process of writing this book. Her enthusiasm and perspective were invaluable. I received an amazing amount of feedback from first-draft reviewers: Carey Boaz, Karl Brophey, Brandon Campbell, Vik Chadha, Mauro Cicio, Mark Donoghue, Pat Eyler, Ben Goodwin, Jacob Harris, Adam Keys, Steve Morris, Bill Nall, Wesley Reiz, Avik Sengupta, Kent Spillner, Sandesh Tattitali, Craig Utley, Greg Vaughn, and Peter W. A. Wood. They truly made the book better, and I can't thank them enough for their time, energy, and insight.

The ideas in this book were inspired by the many great people I've had the opportunity to work with, both officially and unofficially, over the years. For listening, teaching, and talking, thanks to Donnie Webb, Ken Smith,

Walter Hoehn, James McMurry, Carey Boaz, David Alan Black, Avi Bryant, Rich Kilmer, Steve Akers, Ali Sareea, and Jim Weirich.

Thanks to the all extended family that adopted us in India, and especially to Suman Nag, Ramesh R., Bharath Kalyanram, Sheela Singh and Singh Ji, Rupali Wadhi, Fayaz Uddeen, A.K. Sreekanth, Brenda D'Souza, Vishal Kapoor, and Deepa Rajamani.

Thanks to my parents for their constant support. And most importantly, thanks go to my wife, Kelly, for making this all worthwhile.

The Pragmatic Bookshelf

The Pragmatic Starter Kit series: Three great titles, one objective. To get you up to speed with the essentials for successful project development. Keep your source under control, your bugs in check, and your process repeatable with these three concise, readable books.

Facets of Ruby series: Learn all about developing applications using the Ruby programming language, from the famous Pickaxe book to the latest books featuring Ruby On Rails.

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style, and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help programmers stay on top of their game.

Visit Us Online

My Job Went to India
`pragmaticprogrammer.com/titles/mjwti`
This book's home page, including errata and other resources.

Register for Updates
`pragmaticprogrammer.com/updates`
Be notified when updates and new books become available.

Join the Community
`pragmaticprogrammer.com/community`
Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy
`pragmaticprogrammer.com/news`
Check out the latest pragmatic developments in the news.

Buy the Book

If you liked this PDF, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragmaticprogrammer.com/titles/mjwti.

Contact Us

Phone Orders:	1-800-699-PROG (+1 919 847 3884)
Online Orders:	www.pragmaticprogrammer.com/catalog
Customer Service:	`orders@pragmaticprogrammer.com`
Non-English Versions:	`translations@pragmaticprogrammer.com`
Pragmatic Teaching:	`academic@pragmaticprogrammer.com`
Author Proposals:	`proposals@pragmaticprogrammer.com`