

Preface

I know why you've picked up this book.

You're looking at it because you need to work with programmers in some professional capacity, and you're nervous and a little intimidated at having to work in such an exotic environment and with such strange people. You're wondering whether you're competent to do the job or whether you should turn it over to some expert with more experience.

Relax.

If you're a professional coach, counselor or consultant who works in the business world you already have the skills you need to work with programmers in the software world. What you're missing is a knowledge of the differences you should expect, and that's precisely what this book is going to give you. When you're done you won't be an expert by any means, but you'll have the background you need to walk into the new job and understand what's going on.

When your client tells you, "Those damned product managers are trying to slip another requirements change past us. We're going to have to rewrite the design specification and completely rebuild the user interface!" you won't panic, you'll say to yourself, "Okay, I know generally what she's upset about - at least enough to go forward with." and then you can start doing your job.

You can read this book in any of three ways, depending upon what you need to get out of it. Since it's fairly short, you can simply read it from start to finish to have the topics presented in a logical order; you should be able to finish it in a day. But if you need to

know something about a specific element of how programmers live and don't want the panoramic sweep, I invite you to use the Table of Contents to skip to the bit that you need, because I've tried to keep each chapter fairly self-contained. And if you're really in a rush, you can read the summaries called "For Those In a Hurry" at the end of each chapter. These will give you the absolute minimal sketch of the content chapter you need to go into a meeting and not look stupid. But in any case, you should eventually get around to reading the whole book, because despite my best efforts, the chapters are interrelated and you'll have a much better knowledge of your favorite topic if you can relate it to the rest of the materials.

As you'll see, this isn't a scholarly book. I will make a lot of claims on my own authority that, in other situations, would require references to studies and scholarly analysis. Since I'm asking you to accept my word for this knowledge, you should know something about me. I started programming in 1967, when computers were still being called "electronic brains" in the media. I abandoned an unpromising career in English Literature to pursue a Masters of Arts in Computer Science from Duke University. I started programming professionally in 1973, and I've wandered through much of the landscape of the computer industry: I've worked in very large companies and small startups; I've done contract work for the US Air Force and for IBM; I've been a low-level coder, a manager of Quality Assurance (briefly), a system designer, a Team Leader, a Project Manager (briefly), and finally a senior software architect. My final gig was designing an enterprise scale employee time and attendance management system using J2EE technology. You don't need to under-

stand what that means, but you should be just a little impressed.

When I got bored with building software I got a Masters of Arts in Community Psychology and a Certificate in Job Stress and Healthy Workplace Design, both from the University of Massachusetts at Lowell, then got certified as a life, career, and executive coach. And now I provide career and executive coaching to programmers and their managers and try to straighten out programming organizations when they get into trouble. I'm not really trying to impress you (well, maybe a little) but I am trying to suggest that I've been deeply embedded in the world that I'm trying to describe and I understand an awful lot that will seem very confusing to you, and know how to explain it in your terms. Think of me as Virgil leading Dante through the Inferno.

It's true that you will be getting my opinions and observations on many of these subjects, so you should take what I say as as guidance rather than gospel. Make your own observations of what's going on around you and, if they don't agree with what I've written here, trust your own intuition and move ahead. And above all else, enjoy yourself - the world of the programmer is usually confusing and chaotic and frequently stressful, but it is always fascinating and entertaining.

Chapter 1 - For All You Civilians

If you don't mind, I'm going to call you and your colleagues who are not professional programmers, "civilians." This isn't a term of disrespect and it doesn't demean your skills: it's how programmers refer to people who aren't steeped in the mysteries of software development and might need a little more explanation - which describes you perfectly. But let me be a little more clear about the professionals who might find this book useful and how you might benefit.

Professional Coaches

The software industry was among the last to embrace the use of executive coaching to help managers cope with a very demanding job, and you'll still encounter the rugged individuals who feel that a good manager should be able to cope with any situation by herself and that coaches are just for New Age sissies.

Programmers are, above all, problem solvers; and they don't like to admit that they might need help dealing with issues that are outside their usual analytical abilities. Programmers most need a coach when they're facing a problem that doesn't yield to analysis, and need to use other parts of their personalities. Unfortunately, this is when they're most likely to dig in and offer resistance to your coaching.

You'll have to put up with this attitude for a while, but there's growing evidence that coaches help executives, middle managers and even programmers in the software development world. In the meantime, your work will be more effective if you understand the personalities involved and the things they do for a living.

Mental Health Professionals

If you're a Psychiatrist, Psychologist, or Mental Health Counselor who is working with one or more professionals from the world of computer programming, there are three things to keep in mind. First, you're going to have to shed the stereotypes that label programmers as cold, introverted, left-brained intellectuals. As you'll learn, there's a certain persona that programmers put on when they walk into the office, but you're really dealing with a person much like the rest of your clients. It may well be that your client is letting the programmer persona take over in inappropriate situations, but you shouldn't have too much trouble making contact with the person behind the mask.

Second, it's very likely that your client's problems are caused by or exacerbated by episodic stress, because she is frequently called on to perform miracles within very compressed schedules and she may be exhausted and burned out. There is a code of machismo in the software industry, expressed in the archetype of the "hero programmer" who can work for weeks at a time with no sleep, living on Twinkies and Jolt Cola. This is, of course, nonsense, but you'll have to contend with it as you help your programmers learn to take care of their mental and physical health.

And finally, be ready to treat a very intelligent client who may understand the dynamics of therapy and may in fact stay several steps ahead of you in the relationship. You may have to help her through the less analytical activities like visioning, role-playing, and trusting her emotions, but she'll understand what you're trying to do and can participate in planning her own therapy. You'll probably enjoy the experience.

Employee Assistance Professionals

EAP professionals will have to face the same challenges as do counselors, but you will be much more embedded in the company's culture. Since you'll be seeing clients who are fresh from the battle front, you can expect them to be frustrated and stressed, and to spend a lot of time in story before they can get down to solving their problems.

And because you're paid by the company, you may also face skepticism that you're working for the client's best interests. In short, you won't be facing anything that you haven't encountered before. But you'll also be working with intelligent, motivated, high-performing clients who will respond well to short-term therapy and can participate in planning the next steps of their own recovery plans.

Organizational Psychologists

Organizational Psychologists are likely to be very confused by the software environment, especially if they are used to working in hierarchical command-and-control organizations like insurance companies or manufacturing. The world of the programmer is characterized by loose networks of individuals and teams and the flow of real authority through the organization can require a map to follow. You should leave your current models behind and spend a lot of time just watching what programmers do, whom they talk to, and whom they defer to - then you'll start to get a feel for how the organization really works.

One other thing - if you're used to relying on assessment instruments to measure personality or aptitudes, leave them behind as well; it's very likely that pro-

grammers will understand how to manipulate them before they're halfway through the administration. Instead, learn to trust your instincts about the programmers, how they relate to each other, and how they respond to management. Get ready for a wild ride.

Guidance Counselors

If you're a college guidance counselor and a nervous Senior asks, "So, should I take that programming job?" would you be able to give her any meaningful advice beyond salary range? To answer her question you'll need to know more about the kinds of people she will meet, what she'll be expected to do, and what triumphs and frustrations she's likely to meet. If you have an opportunity, try to spend a few days in an active software development project just looking around and asking questions. Then you can match this information to what you know about your student and give her a confident answer to her question, even if the answer is, "You'll have to check it out for yourself."

Human Resource Professionals

HR professionals: either you already work in a company that develops software and you realize how chaotic it is, or you're going to work in a software environment and you're worried about how to understand what's really going on. You'll probably be okay if you treat the programmers as individuals and don't try to generalize too much about them.

You won't have any unusual troubles with benefit administration, but it may be very difficult indeed to write accurate job descriptions, and you'll find that

performance evaluations will be very subjective and subject to a lot of argument. Also, you'll find that the notion of a career path means very different things to programmers than you're used to.

But the people you meet will be intelligent and interesting, and can participate effectively in their own career planning. And, although it will sometimes be a frustrating job, I can promise it will never be boring.

Organizational Developers

If you're helping a software development organization with cultural issues or helping it implement organization-wide changes, you should start by reading the recommendations for Organizational Psychologists: some of your tried and true techniques just won't work in the software environment because programmers are smart enough to figure out the intervention, or how to bias the surveys, or to skew the performance metrics in their favor.

You'll do just fine if you remain flexible and put your faith in what you see around you, rather than the way things should work. Remember that the configuration of teams, projects, loyalties, and authority will be shifting moment by moment, and be ready to dance along with it.

Spouses, Partners, and Families

If your kids ask, "What does Mommy do at work?" you may have a hard time answering in any coherent way. Indeed, she herself may have difficulty describing a seemingly random process in terms they can understand. You'll have a much better chance of following

along if you know the cast of characters and some of the basics of software development.

Maybe your spouse comes home at around 10PM and seems dejected and downcast. You ask him, "You're home so late - what happened?" and he just shrugs and mutters "Well, our component won't be ready for system integration, so we're holding up the beta test." If you knew just a little more about him and his world, you could see this for what it is: a passing crisis that demands some more hard work and long hours, but will eventually pass.

Interested Bystanders

You may not fit into any of the categories above - you might be in another branch of the helping professions, or you might be a manager trying to understand the crazy people who work for you, or you might even be a programmer who's trying to understand the chaos going on around you. You might simply be interested in what happens in those strange buildings that you drive past, or what those strange people in the coffee-shop are talking about. Whatever your reason for picking up the book, I hope that you find it informative, useful and, above all, entertaining.