# EXCELLENCE IN IT

# EXCELLENCE IN IT

## Achieving Success in an Information Technology Career

WARREN C. ZABLOUDIL

Universal-Publishers
Boca Raton

*Excellence in IT:*
*Achieving Success in an Information Technology Career*



Universal-Publishers
Boca Raton, Florida • USA
2014

ISBN-10: 1-62734-025-4
ISBN-13: 978-1-62734-025-0

www.universal-publishers.com

Cover image @Cutcaster.com/grafikeray

# TABLE OF CONTENTS

## CHAPTER ONE

# Necessary Assets

No one is born naturally good at technology. Natural-born techs are a myth. In fact, Information Technology, or IT, has nothing to do with human nature. Yes, it's a product of humanity, and yes, IT is part of our modern culture, but machine based computation was never part of the human condition. It's simply a tool of convenience recently added to man's repertoire and nothing more. There's no throwing, hunting, or tackling involved in IT work…at least physically speaking. However, some nonetheless see tech work as a mental exercise not all that different from chipping a piece of flint into a spearhead. They argue that since it also involves problem solving, Information Technology taps into basic human nature and thus a person can be a natural at it.

The truth is that so much of person's mind is developed after birth that it's impossible to say how many problem solving skills were nature given and how many are developed through life lessons. What's more, the sensory skills needed to make a good flint spearhead are largely removed from the purely mental act of working with computer systems. You can hear weakness in a piece of flint by tapping on it. Touch, strength, and a fairly extensive amount of physical coordination are also part of making a first rate spearhead.

Although the spearhead maker may have visualized the finished product before he began working, just as modern system developers do with solutions they're working on, the act of turning his mental image into a material possession was primarily a physical act with some measure of problem solving thrown in along the way. It's a vastly different and much more advanced act when you turn a visualized image into a virtual outcome. A virtual outcome is measured only by an abstract input and output, which creates a product that is seen but never touched…at least not directly.

In contrast to the idea of a natural-born tech, practically all complicated jobs you'll inherit while moving up the tech food chain will involve skills you learn along the way. For instance, multitasking is a skill-set you'll definitely need to develop to stay in the game if you didn't possess it beforehand. Not just run-of-the-mill multitasking either. If you're working on more than one job at a time, then it's

not merely balancing several unrelated tasks at once. You'll find yourself moving from one *group* of tasks to another in a non-linear order. That means working on task two from job B, task four from job A, tasks one and three from job C, and so on, all at the same time. Experienced techs already know all about this. You must move quickly and easily between the different tasks in different jobs without losing the perspective of each task's individual role within their respective job. Sound difficult? It's just a matter of practice. Anyone who wants to excel in IT must be good at multi-tasking multiple-tasked jobs. This can only come through repetition and experience. The ability to simultaneously track different things with multiple parts is a modern, learnt skill, and not just a part of our shared human nature, so quit pretending there are natural born techs whose skills you can never match. Everybody starts at the same place. It's just matter of how hard you work at being good at what you do. The bottom line is that everybody starts at the same level when they decide follow a career in information technology.

If there's no such thing as a natural tech, there's also no such thing as the hopeless case who can't get the hang of things either. It's always just a matter of personal effort. How much effort you bring to the job everyday defines how good you'll be at your job over time. The effort to be a good tech can be made easier by adopting four simple attributes that will help you along the way. Work to maintain these attributes every day and your career path will become easier.

## The Four Attributes of IT Excellence are:

- Courage
- Focus
- Clarity
- Sense of Scope

Aspiring to these four attributes will help you enjoy a long and happy career in the world of computers:

## Courage

It's safe to say that every tech will have a moment of hesitation at least once in their career when asked to take on a critical job. This's especially true when the job involves a leadership role in an area that affects many people in the company. The company's productivity will be at risk as well as the tech's credibility. If anything goes wrong, there's no place to hide. Your coworkers will know who was responsible for their inconvenience. That kind of pressure can make even the best, most experienced techs pause. Nevertheless, modern computer systems are online and running successfully all around the world, so somebody must be finding the courage to build them. The truth is that good techs have been finding the strength to build complicated and important things for many years now regardless of how critical the systems are to the people around them. Once they got started, those techs then found the confidence they needed to follow the job through to the end.

To be clear with the definitions here, courage is the ability to accept a new challenge while confidence is the bearing you maintain while the challenge is underway. The two are not quite the same so don't mistake one for the other. Don't ever think that solely one or the other will be enough to get you through a big job. Courage without confidence is starting a race you won't have the fortitude to finish; confidence without courage is being well prepared to follow a path you'll never take. Courage gets you started and confidence keeps you going. Both are only useful together.

It can be overwhelming at first to be given a tough job which affects many people who you know personally and work with every day. Take heart in the fact that there are ways to minimize the weight of the task. The first thing you must do is develop good preparation skills. For example imagine a crazy guy who, on a dare, is about to jump off the two-story roof of his house into his backyard. Just before he jumps he'll probably be more nervous than another guy with a parachute on his back who's getting ready to jump off an eighty-story building at the same time. That's because of the amount of preparation each jumper undertook. The two-story guy probably won't die; he'll just break a leg or two. What he's about to do has nowhere near the stakes of the eighty-story building jumper. Nonetheless he'll be more afraid than the building jumper at the moment their respective leaps are taken…as he should be.

A life truth is that preparation is the biggest part of courage. This applies to everything you do and goes double in the IT world. Nothing affects your courage more than appropriately preparing yourself ahead of time for a difficult job. Skill-set management, planning, and sufficient testing will all make a positive difference in your attitude before you start. If you need to take some time to review the technology first before starting on new task, then take the time needed (within reason of course). If your boss dropped the job into your lap while adding that it "needs to be done yesterday," then you need to get those communication skills going to convince him that "needs to be done yesterday" and "doing it right the first time" don't go together. If that doesn't sound easy, so be it. Do it anyway. Preparation is a large part of your professional demeanor and your professionalism is one of the few constants in the constantly changing world of IT. Preparation gives you the courage to move forward when everyone else is holding back. As skill-sets come and go, as experience on obsolete systems fade into the past (trust me, the day will come when you really can tell the new guy you've forgotten more than he knows), and as employers go through corporate changes, you must cling to your professionalism like a life preserver. It's the one thing that will keep you proud and confident. It's also the one thing that will keep you satisfied over a long career in computers. When you lack courage, your professionalism is at risk. Everything you do is vulnerable to being compromised by stronger willed people around you. Never let this happen. If you feel pressured or bullied or kicked about, remember it's the systems you build and maintain that have the final say in how good a tech you are. Build them well and keep them running and only good things will follow. No one can take away up-time from you and up-time (when the systems run well) is the only true measure of how good you are.

Still, there is the occasional worst case scenario. Let's say for example there's a rushed boss who wants you to jump into a job you know you're not quite ready for. He wants you to move on the spot and you, as a professional, prefer to take a moment to get up to speed on the technology first. The boss is so disappointed in your lack of recklessness that he finds someone else to get the job done on the double quick. To make matters worse, when the new guy takes over he gets lucky and finishes the job correctly and on schedule too. As a result, the boss starts treating you like a goat and the other guy like a hero. If this happens to you, then keep one critically important thing in mind: *you were still right*. Even if the new guy

succeeded brilliantly, he was wrong to begin in the first place and relied far too much on luck to finish the job.

To pick up on the previous analogy, let's say the guy who jumped off his two-story roof hit the ground and rolled to a stop uninjured. He pops right up and is the hero of the moment, receiving "oohs" and "aahs" from everybody around. Does that mean he can climb right back up and do it again? No. What it means is that he made a bad decision and got lucky. The next time he jumps, he faces the same tall odds of disaster. The same goes for any tech who rushes into a job without suitable preparation. If fact, pity the tech whose boss is so impressed with his lack of restraint that he's now expected to get jobs done 'on the quick' every time. That's no different than expecting the guy who jumped off the roof to keep doing it over and over again on a regular basis. After all, it worked out fine the first time. If you're professional enough to want the job done right instead of just quickly, the fact is you were correct in your restraint. Your boss, who was cutting corners, is only inviting trouble down the road for the whole department.

Doing things without proper preparation will always cost the company extra money in the long run. The inevitable mistakes that kind of work ethic brings will cause more downtime and delays in worse ways than taking time to properly prepare ever will. If you find yourself in this situation, remember your boss made a bad decision based on the needs of the moment and you made a good decision based on your long-term professionalism. Don't ever doubt yourself in this regard, no matter how much it may sting your self-esteem at the moment. In fact, it's even reasonable to think that if the boss had planned things better from the start the current rushed situation wouldn't have happened.

Good preparation means understanding all aspects of the job at hand. This includes end-user requirements, the technology involved, the milestones to be managed, the risks during implementation, and the abilities of those who are assisting you. The more you can lay down a clear path to follow in terms of rolling a system into production, the more courage you'll have from the beginning and the more confidence you'll maintain along the way. Good preparation leads to courage, courage invites confidence, and confidence leads to success.

Once you understand the technology enough to proceed with a reasonable amount of comfort, the next step is to measure all the variables. These can include such things as cost, workload, time, risks, workmanship, and user orientation. The combination of

variables is different with every job and you must work through them all to determine their importance in each case. Confidence can be gained by understanding how much focus each variable deserves relative to its impact on that particular job. This is because understanding something in detail helps diminish the element of surprise and, as has been proven over and over again, surprise is the one thing that can never exist in the world of IT.

Being able to understand all the variables in depth for each job is something that comes through experience. It's never a quick task. You must take care to understand as much as possible about how to proceed on a job before you start to work. Never think that you'll figure things out as you go. Planning to "cross that bridge when you get to it" doesn't work in IT. Presume nothing and prepare for everything. If you still get tripped up along the way, you'll at least be able to recover rapidly and get things back on track with less effort.

The variables that can cause you to stumble during your work are measured as risk. Risk comes in so many shapes and sizes you may not even realize it's even there until too late. It could be a loss of funding caused by a previously unannounced change in corporate structure, a component stuck on backorder with a vendor, a team member overstating their skill-sets, or a large end-user group struggling to find time for proper orientation once the new system is successfully built. Risk can come at you from every angle and at any time along the way. Being burnt by unknown risks can dampen your courage for future jobs and hurt your confidence while finishing the one at hand.

It takes imagination to understand risk. The more ways you can imagine how things can go wrong, the more prepared you'll be before you start. Minimizing risk is the same as minimizing surprise. Don't let yourself be surprised by anything as you move forward. It's not good enough to try to merely anticipate risk. You must fully understand and eliminate as much risk as you possibly can before you begin. An IT project must not be an adventure into the unknown. There can be no unseen elements hiding around unforeseen corners waiting to trip you up. You must anticipate where all those corners are and what elements of risk they might conceal before you begin.

Nothing gives you more courage than knowing how something will turn out before you start. One of the nicest things about the IT world is that you get to work on things you control from the beginning. After all, those systems didn't fall from the sky. They were built

either by you or someone like you. As such, you have the power of God over what goes on inside them. If you don't fully understand them, don't start working on the job until you do. If deadlines that don't allow time for understanding were irresponsibly assigned by management, then convince management more time is needed. Running head-on into a fail-state that could have been foreseen with a reasonable amount of preparation can cripple a company and break morale. No matter how big the hurry, unforeseen risk can easily end up costing more in the long run than using an appropriate, evaluative approach.

You gain more credibility from a successful job than from a failed one, no matter how heroic your efforts were. In IT, you have the opportunity to foresee the future, at least as far as any particular job is concerned, so don't waste it. Take the opportunity to understand all the elements that can go wrong before you begin so you can move forward with all the confidence in the world. You can predict all events and vanquish all worries ahead of time if you put in the effort to do so.

Bigger jobs may also have more than one deadline. Understanding how risk affects each of those deadlines (or milestones) is a critical part of performing any job well. If the job is multifaceted, then not meeting a particular milestone can be more than just a scheduling hassle. The entire proverbial assembly line can be stopped if one of those milestones is missed. That means even the possibility of a missed milestone here or there is just one more thing that needs to be accounted for in the planning stages. It's especially true if the deadlines were poorly thought out by the staff member who assigned them. The best thing about moving from milestone to milestone is that it gives you a clear line-of-sight path to the next endpoint. If you break even the most unreasonable deadline into manageable steps, you can work your way through it with less effort than if you had simply dived in and hoped for the best. Most importantly, you'll reduce the element of surprise while working to get the job done.

Risk loves things like rush jobs and short deadlines. Those are its easiest targets. Just as water always follows the path of least resistance, risk always follows the path of least preparedness. An important skill-set you should develop is the ability to understand how to manage the risk of those occasional short deadlines. If you get a short notice job and you understand the accompanying importance and the risk involved, the first course of action you must take is the preemptive act of building up the courage to request some

game rules as soon as you can, even if nothing particular has been assigned my management yet. Rule number one is that is that you'll always be given a reasonable amount of time to develop a workable plan for any new job. Rule number two is that you won't be thrown into a job cold with no skill-sets for the solution you'll be working on.

Usually these game rules can be a simple request to your boss. Bosses with any IT sense will appreciate your forethought and probably have a pretty strong idea of what you're requesting based on their own experience. Confidence, as well as courage, can also be handed to you through good management too. If you have a lousy boss who doesn't get it, the best you can hope for is good luck with the jobs you're assigned as you struggle to mitigate risk more effectively. In some cases, it's not unreasonable for you to consider mustering what little confidence you have left and use it to find employment elsewhere; after all, your ability to maintain your professionalism must *always* come first throughout your entire career.

Keep in mind that even the worst bosses can't hide for long from the high employee turnover rate they cause. Turnover rates cost the company money by making it difficult for the IT department to operate efficiently. The hard truth about IT is that no department can cut more deeply into a company's bottom line than Information Technology does. IT is always considered overhead; even in companies that sell IT services. IT doesn't generate a single penny of profit for any company, its true value can only measured by how cost efficiently it can support billable minutes for the rest of the organization. All computer related inefficiencies will eventually cause the company to lose billable minutes and suffer increased overhead. No matter how politically connected the bad IT boss may be, this truth will always catch up with them.

However, what if the project assignment is a 'one-off' anomaly? Let's say a good boss got a rush request from a customer that can't be denied. Then you need to figure out how to professionally mitigate risk in a hurry. Step one is to take a brief moment to go on the record by politely reminding everyone for the umpteenth time that this's not how things should be done and to truthfully establish an understanding of the increased risk you'll be facing. After all, it's better to get the gripes out before you begin than it is to complain later on when the deadline is approaching and things are crazy enough on their own without you adding to the noise.

Whether you're rushed or have plenty of time to work with, a handy trick to identifying all those pesky risks in any job is to narrow your view to as granular a level as is reasonably possible. The more granular you have time for, the better. Do this after first gaining a solid understanding of what needs to be done and laying out the job milestones as usual. Then divide each milestone into several smaller projects with contiguous start and endpoints. The smaller, more granular view provides a clearer evaluation of risks relative to each milestone. As each risk is identified it needs to be assessed against the scope of the job as a whole. Used properly, this trick provides a more thorough approach to risk identification by narrowing the scale of each part of the job that any given risk can affect.

Narrowing the scale not only helps you better assess risk, as a side effect it also forces you to become more familiar with the job down to the granular level you've chosen. Focusing on the subdivided parts before you begin will give you a more detailed overall picture of how to proceed. This increased clarity helps identify even the most subtle of job-related details and allows you to eliminate any risk hiding in those details before the work begins.

The more you understand up front, the more courage you'll have when you start. The more courage you gain, the more prepared you'll be to move quickly to deal with issues as they arise. This is where the confidence part comes in. Even those elements of risk that managed to remain invisible during the planning stages won't be quite as overwhelming when they rise up and try to knock you off schedule. No matter what happens after you begin, the more prepared you are at the start, the more confidence you'll have in the middle of the job in the event any setback does occur.

Preparedness is the result of both planning and training on the plan. For preparation to be considered fully complete, at least one rollback option must also be in place in case something goes wrong beyond your control. A rollback plan is not only a critical step to ensure the operational continuity for end-users; it's a real confidence booster as well. A rollback plan is the ability to turn the clock back to a time when everything was still functioning properly. It's the reset button in the video game of life. Rollback is made up of the contingencies needed to keep a failed implementation from ever reaching the end-users. A good rollback plan isn't based around a single trigger either. Rollback can be total if the implementation was a complete disaster or just partial if the implementation went mostly well, but had a few bugs along the way. Every undertaking is always

less scary if it includes plans for how to get back to a safe place if need be.

A final note on courage is that it's almost a certainty that every busy IT professional tech will experience at least once in their career a bad spell in one form or another when anything which could possibly go wrong with their work usually does no matter how hard they try to make it right. Even the best techs out there can experience times in their career when it appears fate is against them, or at least extending some persistent doubt in their direction. This type of thing usually happens when you're going through a "snake bit" period. That is, while everything you touched before on the job seemed to turn to gold, now everything you touch in your work seems to turn to lead. You're still a great tech, but, for some unknown reason, no matter how good you always are at making judgment calls about what to do next, for no apparent reason those calls simply start going wrong despite your best effort. When you look back at your recent work all you see are mistakes…usually dumb ones too.

This happens to everyone every now and then in the complicated and constantly changing world of computers so don't take it personally. If you're doing your best and your best is good enough, then there's no explanation for being snake bit that'll make sense so you shouldn't let it hurt your confidence. Just keep doing your best and the rest will take care of itself. It's an honest truth that every tech in the world will experience a snake bit period at least once in their career; including you. When you're snake bit, even your best work can leave a dirty trail behind you. You look back at your recent jobs and all you see are problems all around. Not big, complicated mistakes either, but irritatingly simple ones you can't believe you keep making. Since this kind of thing happens to everybody at least once over a long career, you should create a "snake bite kit" ahead of time for whenever the situation finally arrives. For starters, remember that being snake bit is always temporary. It's best to just buckle-down and push on through it. Soon enough the successes will start returning and your courage and confidence will get back on track. You can, and should, bank on it.

One tool that should be in every snake bite kit is your resume. It's old advice, but as long as you know you know it's truthful, re-reading your resume helps remind you that your self-confidence is not unwarranted. You did good work in the past and you'll do good work in the future too. Revisiting your past successes will often give you the boost in self-confidence you need to move forward with

your head held high. As long as you know you're good, the successes will return. When they do, count them…literally. Always keep your resume up to date if for no other reason than to remind yourself that you are a first rate tech. That kind of continuity in your confidence over time is what will help you rise to the top of your professional world and earn a reputation for excellence in your company.

Other curative steps can include reviewing your certifications or degrees and what you did to get them. Also, any past accolades from bosses or coworkers are worth review too. This's a highly personal act so design your snake bite kit as you see fit. The thing to remember is to never lose confidence in yourself, even if it others around you appear to have begun to doubt your ability. It's OK to give yourself credit even if no one else does. In fact, it is a critical part of moving over those rough spots in your career and keeping your courage level high at all times.

## Focus

Nothing defines a solution better than the number of details it involves and nothing measures a tech's skills better than their ability to work with those details. Having good focus is how you come to terms with those details. Taking solutions from a variety of vendors and combining them into a single system means adapting the nuances in each of those solutions to work alongside all the other nuances around them. Being able break something down into its basic parts so you can accurately focus on its details is a critical skill for anyone who works with complex elements for a living. In reality, the old adage, *keep it simple stupid*, only applies to jobs that were simple to begin with. The beauty of a great solution will always be found in the complexity of its details.

Many techs have a problem coming to terms with the true level of detail that good preparation entails. Those techs need to learn the hard way that overlooked details will always come back to haunt them when moving to production. You should guard against this kind of education. Don't let this happen to you. Take the time to inventory all the details involved with the job and touch on *each one* before starting to work. Maintaining a high resolution focus goes a long way toward achieving excellence in the tasks you're involved in. For a tech traveling to a site on a service ticket, this could mean going through all the possible causes of the problem in their mind before they arrive. For someone starting a new project, it means

working through the project and accounting for all the details in a thorough project plan before moving to development.

The more you trial your solution to account for all of its details before moving to production, the better off you'll be. This's because trials are a great way to expose the hidden problems in your system. Failure is one of the most useful tools you can have when working to get things right, so it's actually a good thing to fail during a trial. The more you understand about how things can go wrong while you're still in the preparation phase, the better everything will work when you finally move to production.

There are different outcomes that people may be inclined to call positive for any IT job. There is the top grade "right the first time, every time" outcome, the bottom grade "good enough for now" outcome, and several shades in between. It should be obvious what kind of outcome you should strive for if you want to be thought of as an excellent tech. Always remember that from your perspective as an IT professional, the definition of quality doesn't come from how well the computer system runs when you're finished working on it; that's the end-user's definition of quality. From your perspective, the definition of quality comes from the number of those exquisite details you accounted for while the job was underway. The rest will take care of itself when the job is done. The more granular your focus is at the beginning, the higher quality your output will be in the end.

If you want to be a real professional, you must always be able to recognize the difference between good work and "good enough" work, even if end-users won't ever notice it. Professionalism in any line of work is gained more through the practitioner's ability to understand the smallest details of their craft more than from anything else. No amount of happy talk will ever replace that. Whether it comes from classroom education, hands on experience, or a good resource library, the ability to maintain an accurate focus on the smallest aspects of what you're working on is what will most define you as being good at what you do. Anything less will just define you as less. It's the good techs who keep track of all those fine details that will develop the reputation of being excellent.

## Clarity

It can't be said enough: the best measure of how well you understand anything complicated is how clearly you can explain it to someone

who knows nothing about it. This goes double in IT. Few things are more mysterious to end-users than their computers and whatever it is their computers are tied to. The gap between the knowledgeable and the novice can never be underestimated when interacting with end-users. What's more, the definition of novice can be much broader than the average tech might think.

If you look at it from the perspective of your daily activities and how they continually create something new in your environment, then anyone, including your boss and your teammates, can be considered a novice, at least with that one thing you're working on at the moment. Your work in IT affects so many people that you're more or less required to keep all involved parties abreast of your actions to some extent at all times. This can be a status report to a boss, a quick answer to a teammate, some updates to the helpdesk, or a notice to end-users about something new coming their way. It can also be a reminder given to upper management about why the expense of something new is worthwhile. While technical skills are part of your craft, you should treat communication like an art form because it is really that important.

The critical thing to remember about practicing the art of communication is that while you multi-task multiple-part tasks all day long, you must always communicate about them serially. That means to sticking to only one topic per answer. When you're giving a status report always follow a linear order and don't jump from one task to another. While all those multitasks make complete sense in your busy mind, your listener is going to become confused in no time. It's amazing how difficult it can be for some techs to communicate sensibly when they're working on a number of different multi-tasked jobs at once. Any status updates they give are basically just a brain dump of barely connected facts.

To make matters worse, the typical tech is surrounded by other multi-tasking professionals, all giving status updates, asking advice, or anything else that involves describing their ongoing activities. It's critical that nobody add more than one topic at a time in this situation. It's hard enough to keep track of your own jobs without trying to follow along with someone else's jobs being described to you in haphazard order. Clear communication starts with simplicity. The thing to *always* do when communicating is to find the simplest, yet fully accurate, method for conveying *just* the information required for the moment and nothing more. Any elaboration should only be used as a last ditch rescue attempt when clarity has failed after a try or

two. Even then, it's critical to be short about it. When elaborating, for whatever reason, remember that if you weren't clear in the first place, adding a lot of marginally organized detail later on isn't going to help the listener much. Master the art of saying just enough to be clear, concise, and accurate with your information from the start. It'll serve you well in the long and make your listeners happier, too.

The same goes for what you write. Keep written communication short and to the point. It's easy to get in the habit of putting everything into a single e-mail that leaves nothing uncovered. This can be especially true after spending time with those "but why?" end-users who persistently e-mail questions about things in a level of detail you know they don't fully understand. As soon as you answer one question, they reply with another that's equally hard to quickly explain. Pretty soon you have a long thread running down the screen of back and forth "but why" questions and answers.

Sometimes techs respond to this type of end-user by preemptively including a response to every anticipated question in one big "first strike" e-mail that covers every possible detail. Their first strike e-mail usually ends up being a bunch of paragraphs that run on forever. While this's better than blowing off the overly inquisitive end-user with "JUST BECAUSE" in bold font, it can lead to the bad habit of responding to all e-mails with a first strike format. Don't let this kind of thing change the way you answer questions in general or you could find yourself putting everything into one big e-mail even when no one is asking for it.

The irony here is that when the recipient sees all your well-chosen size eleven Calibri font words running down the screen, they may not even bother reading your e-mail in the first place. Nobody likes reading long e-mails in the middle of the day. If you have ever had someone ask you a dumb question just after you explained everything to them in an detailed e-mail, don't blame them. It's not their fault if your e-mails are too long to read. It's best to keep to short answers and avoid the long novels. While you're at it, break your e-mail up into small blocks too. Put an extra line feed in every now and then. Even when a paragraph is relatively short, it can still look too long to read if it's a single block of text. Remember to never be a hassle. You're not the only one who gets tired of reading e-mails all day long. Add a little extra formatting to make your communication easier to read. Your recipients will appreciate it.

Good communication skills don't only apply when interacting with others; they're also important when communicating with your-

self. This isn't meant to be trite. Being truthful with yourself makes a big difference in how good a tech you'll end up being over time. This isn't about being hard on yourself: self-honesty has nothing to do with marginalizing your own self confidence. However, being able to communicate truthfully with yourself means understanding your limitations in both knowledge and time management. All the knowledge in the world won't help much if you never have time to use it. On the other hand, having only a few skills-sets but a lot of time to figure things out may seem nice, but you won't be getting much done. The bottom line is; being truthful with yourself will help you make better decisions about the jobs you're working on.

This particularly applies to knowing when a job is truly done. This is one of the hardest lessons for techs to learn and is the biggest failing of rookie IT professionals. For example, stress can create a deep desire to just tip-toe away from a quick fix in the hope that the problem won't pop back up the moment you leave the area. Or a tough time management issue might have you running off to the next job before the current task is really finished. Sometimes less experienced techs don't know enough about what's being worked on to feel comfortable with sticking around to confirm that the work they implement will take in the long term. As every tech eventually learns, calling a job done too early will lead to repeat visits to work on the same issue down the road. The worst side effect of this is the end-users will eventually start to question your abilities.

Learn to deal with this situation by communicating with yourself well. First, listen to that frustrated voice in your head that tells you that this time you really will crack down and study what's needed to be proficient on the systems you're responsible for. When you ignore that voice, you're ignoring common sense. Get to know that voice and understand what it sounds like. If you want to claim a good level of street smarts on the job, it's your common sense that'll give them to you. Ignore common sense and you could find yourself both miserable and unreliable. Many IT professionals, if they give themselves a chance, can be surprised at how often their first notion turned out to be the correct one and how much it helped them strive into a new job with maximum effect by allowing them to move quickly from the beginning.

Striving into a job helps you to make a better call about when the job is truly done. In fact, it's the biggest part in getting things done right "the first time every time." If you want to gain excellent results with the tasks you're given, you must always listen to your

common sense when it tells you a job isn't quite complete. Always keep in mind that it's you alone who brings the last measure of your work ethic with you wherever you go. Never mind your boss. No matter how much your boss might threaten you, that source of stress only accounts for about 96% of your best effort on a good day. The remaining 4% of effort, the part that makes you an excellent technician, is something you must bring to the job on your own. At some point you must move beyond the simple job order and take ownership of the work you do. It's in the pride of ownership that you find that final 4% of effort. Never settle for the 96% solution, even if it makes your life easier for the moment. Let your common sense give you the clarity you need to cover that final 4% on your own and you'll gain excellence in no time.

## Scope

If you want to be a superstar, or at least the best tech in your department, taking single steps at a time is the best approach to follow…no matter how ambitious you are. If you're addicted to accolades or just like having a say in how things are done, the status of an excellent worker is how you can achieve your goal. The important thing to remember is that you must never overreach or your efforts will end up doing more harm than good. Never lose track of how much you can legitimately do and only push a small step beyond that limit each time you're ready to move ahead in your career.

Taking on a job too big for you to handle or taking on too many jobs at once can damage your credibility if things go wrong. On the other hand, stepping forward to volunteer for the tough jobs on a regular basis is a great way to move your career upward. You must always temper your ambition by taking careful measure of your current ability; otherwise you'll end up biting off more than you can chew. How you balance the need to excel with the common sense of being careful about what you're willing to get yourself into takes some experience. It's all about understanding your reach.

Never mind your physical arms; they aren't part of the reach being discussed here. Instead, try imaging you have virtual arms with the strength of a small construction crane. You can reach those arms way out and pick up very heavy things. However, like any crane operator, you still need to know the limits of that reach or you'll tip yourself over. This limit can be called your professional reach, or more accurately, your extensible reach. Your extensible reach is the

combination of your courage and confidence, your ability to focus, your effort to train hard, your willingness to take the lead when called upon, and what you can actually can do today, all added together. It's the blend of these things when taken as a whole that give you the proper sense of scope when taking on new assignments.

Going beyond your extensible reach will always lead to things which are best avoided. For example, even the best techs can only multi-task so much before they realize they're spending more time moving between jobs than they are on the individual jobs themselves. It's not always a clean jump from one task to the next. Knowing your limits here is critical. If you find yourself constantly stopping to review before you can move to the next item on your list, you're doing too many things at once. It's not that occasional review is a bad thing, because it isn't. However, constant review means you're struggling so much to keep tasks organized that you're losing efficiency. If you're juggling so many multi-tasked tasks that you keep forgetting what each one is about, then it's time to slow down.

Even if it means telling the boss you're too busy, don't feel bad. Going beyond your reach can make you a liability to your department by putting you in the position to make bad choices. Trying to do too many jobs at once can cause you to replace good work with marginal work. It's far better to deal with credibility issues caused by a lack of time or knowledge than it is to work through credibility issues caused by major mistakes.

Some of the blame may also go to managers who aren't allocating resources thoughtfully; however, you shouldn't use this as an excuse. You must find ways to take charge of your workload well enough to make it through each day with balance. Ultimately, it's your career to manage and no one else's. Trying to gain accolades through reckless behavior could lead to mistakes that destroy your chances for dream jobs down the road. Mistakes have a way of following you around, even if you've hidden them from your resume. Have the strength to take charge of your career and don't let anyone, even a boss, talk you into recklessly going beyond your extensible reach.

While the courageous tech will take opportunities when given, the excellent IT professional will only take jobs he is certain he can deliver on. Know the difference between good courage and reckless courage. Overreaching and gambling are both pretty much the same thing; except that in IT you're gambling with your career as well as your department's credibility in the eyes of your company.

A good way to prevent overreaching is augment your current skill-set before taking on a wider range of tasks. After all, there's really no use in being able to do a variety of different things marginally well. Being able to do a few things really well, and then gradually adding to that list of things, is a better long term approach to a successful career. It's also an easier way to handle skill-set management. You should try to avoid being too much of a generalist, even if you're running a small network on your own.

Certifiable skill-sets add confidence and will create success. If continuous skills training seem tedious, that's because you haven't felt the pain and embarrassment that comes from totally screwing something up. Mess up an important system a couple times and all that studying won't seem quite as painful anymore. Stay ahead of the knowledge curve by anticipating as best you can what it is you're going to need to know in the coming year and start running down the learning track early. Not only does this take away much of the pressure which comes from trying to learn in catch up mode, it also puts you in a good position to be one the first to volunteer for new jobs that come along. Preemptive learning is the safest way to push the envelope and extend your extensible reach.

Always begin a job with a strong foundation. Never pretend to have skills that you don't and never plan to learn how to do something after you've already volunteered for it. Both are paths to disaster. It's one thing to drop the ball on something you're familiar with because you can always recover yourself and do it correctly on the second try, but if you're working in an area that's new to you and you don't have solid knowledge about it, then there's nothing to fall back on. There's no deep understanding of what you did wrong to help you recover quickly because you have no real understanding at all.

The best way to stay in front of the knowledge gap is to maintain the scope of your career within a particular area of expertise. When you first start a career in IT, you'll find there are the many paths to choose. You can go into network engineering, application development, WAN storage, software programming, disaster recovery, telecommunications, systems security, systems administration, and so on. Each is different from the next, but within those diverse career areas are even smaller, more specific career paths too. You can broaden out a bit as the years go by if you think it makes you more employable, but try not to broaden your scope too much. If constant learning and knowledge maintenance is tedious for just one area of IT, imagine how difficult it can be to manage constant learning

across several areas of IT. This'll be covered in more detail later in this book, but suffice it to say if you aren't careful there could be a point when you'll have too many things going on with your career to reasonably keep track of.

Most IT professionals find it's far better to concentrate their skills on just one or two paths at most. Being a jack of all trades doesn't work so well in IT. Begin by finding a niche to excel in to start your career foundation and then scale out as you go. If you're new to IT, then this niche might be even handed to you by a boss. Even if this task is not your first choice and you had planned to go in a slightly different direction with your career, do a good job anyway. If the skills don't stay with you after you move on to something new, the professional work ethic you maintain will. Always have a strong sense of ownership with anything you undertake. This includes stepping forward for new tasks only when they're doable in a professional manner. Professionalism comes first, all else follows that.

Even with masterful skills you can find yourself in trouble if you try to do too much at once. It's important to know when your extensible reach is at its limits. Do this by honestly taking an assessment of how complete your 'finished' jobs really are. After all, fixing a tricky bug isn't as easy as changing a tire on a car. Finished and complete don't necessarily mean the same thing in a world where the solutions are often virtual. For the sake of your own long term sanity, you must always resist the impulse to call a job done before it really is. That means get all tasks completely finished by everyone's measure and not just your own.

Leaving incomplete jobs for others to clean up will give you a reputation as the kind of tech who can't be trusted to work on their own. Having someone follow behind you on a regular basis to check your work is no way to achieve a reputation for excellence. Don't fall into the trap of thinking you'll find time to make corrections later either. That's just wishful thinking. The fact is there'll always be more jobs later on that'll take your future time and effort to complete. If you can't get each one completed when you have the chance, you probably won't have time to come back later to finish up. If you reach a point where this's how your workload feels, it's time to slow down and take stock of your situation.

Remind yourself that your professionalism is at stake and that your work ethic should never be compromised. Not doing a 100% job because you think non-techies won't notice your short-cuts is a poor way to work. The discipline that all techs share requires a