THE
Care and Feeding
OF YOUR
Software Testing Career

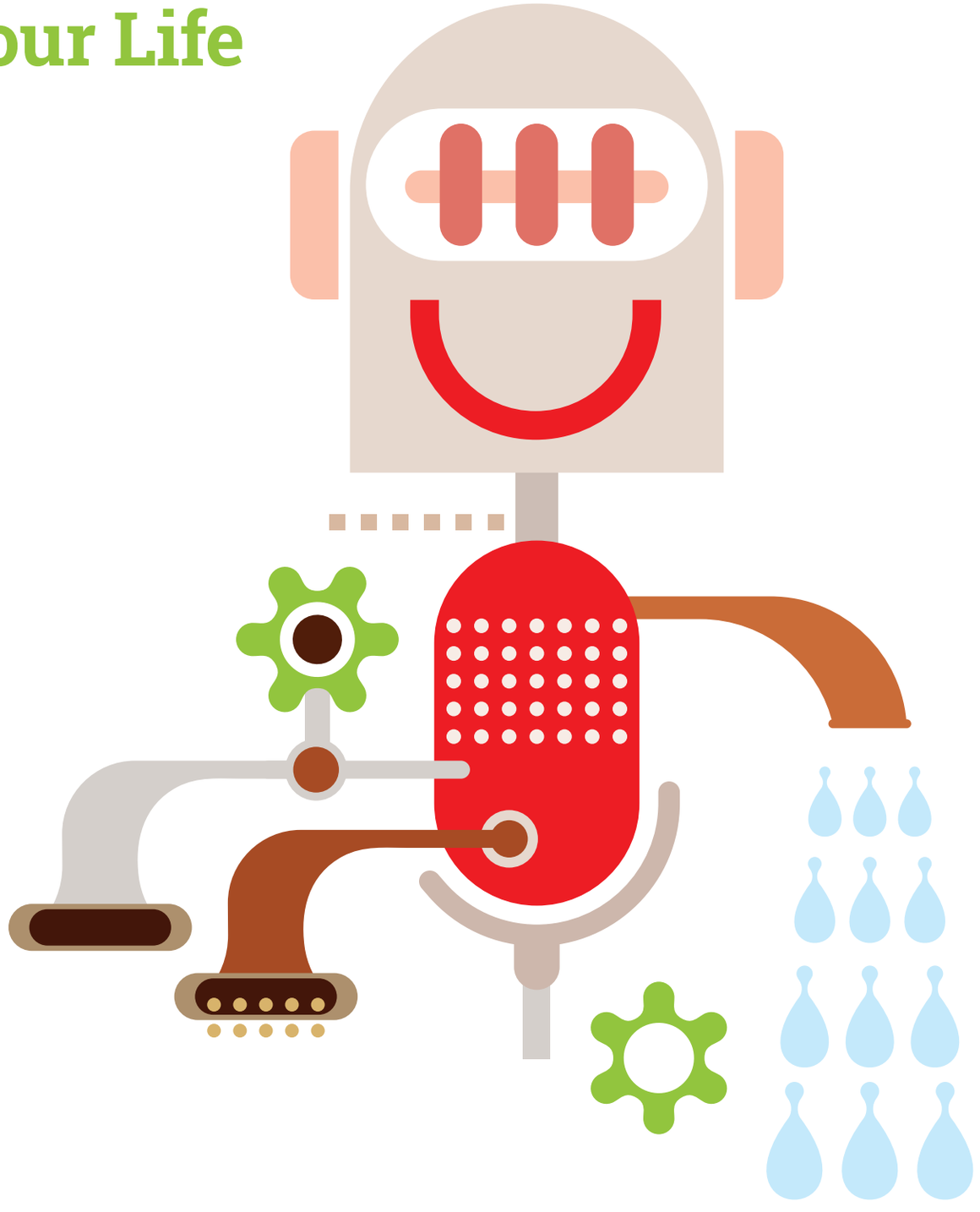
A GUIDE TO THRIVING AS A TESTER IN A RAPIDLY EVOLVING INDUSTRY

BY Michael Sowers

Table of Contents

Chapter 1

Software Testers, This Is Your Life



"What Is Software Testing?" How I Explain What I Do

You're at a party and someone asks you what you do. You answer that you're a software tester. "What is that?" the person asks, and you begin what is usually a much longer answer than the listener can bear. Soon, the person interrupts and says, "Hey, I'm thirsty. Let's get a drink!" Anything to get you to stop all the technobabble.

The work we perform is complicated by the objects (modules, methods, applications, systems) that we work on. Yet, given the ubiquitous nature of software and our passion for our profession, it's easy for us to conclude that everyone understands (or should understand) software.

However, using our software engineering community lingo to explain what we do gets us "deer in the headlights" looks quickly!

I recently read the Bloomberg article **"What Is Code?"** It's a long but good article that uses examples and interaction to describe what code is and how it works.

Still, it leaves the question: How do we simply explain what we do in

thirty seconds? One approach that seems to resonate with those I interact with is to first ask the person you're explaining your profession to, "How many computers do you think you've used today?" After all, any good question deserves a question back, right? Then I remind them of all the places software is present in their lives—their TV, the ATM, cars, elevators, drive through windows, life-critical medical devices, and several hundred other places. Then I ask, "Ever have a problem with any of that technology you use?" Given the high probability of a resounding Yes!, then it's byline time: "I help people develop and deliver better software." If asked for more information, I continue:

> "You see, software is a series of imperfect translations. It starts with someone like yourself needing something, such as displaying your calendar on your mobile phone, and it goes through hundreds of human translations (conversations) and programming (code-writing steps) that deliver you this capability.
>
> "Your need is a requirement. The requirement gets turned into a design, and the design is turned into an application. The application has features that are represented in code called modules, objects, and classes, eventually down to individual characters, and finally bits and bytes that the computer processes.
>
> "A person, usually called a software programmer, develops the code that runs things. Another person, usually called a software test engineer, tests the code that runs things. That's me.

Full disclosure: I seldom get to that third bullet before eyes glaze over!

As software engineering professionals, we each have a role in educating everyone in our circle of influence about just what software is and how critical testing it is to people's everyday lives. Regardless of the approach you take in having this conversation, I encourage you to be an evangelist. In a world run by software, everyone should have at least a foundational understanding of what a tester does.

Own Your Career

One of my great mentors was fond of saying, "Just own it." It simply means to take responsibility and be accountable for your actions, both current and future.

As I survey my career journey, there are times when I've followed this advice, but far too many times I've ignored it and became complacent. This was especially true when I found myself working for several great companies in roles such as tester, testing manager, and even senior vice president of quality and test. Who would want to think about needing to plan your next professional move in the midst of all the challenging, rewarding, and fun work while collaborating with a wonderful group of colleagues and delivering stellar results to customers?

Of course, all of us have experienced both the great and the not-so-great in our professional journeys. On the not-so-great side is when you are being downsized. Having dealt with this twice during my career, I have come to understand the critical nature of always being prepared to move on. I have made a habit of asking myself at least once a month, "If I were

asked to leave or wanted to leave this position today, am I prepared?"

This is so very critical in today's dynamic environment. Gone are the days when companies employ people for a lifetime—or when employees want to stay with the same company forever.

Here's my take on a few of the software testing industry dynamics today.

- As we move from traditional to agile methods, our roles change, and as a result we must acquire additional knowledge and skills.

- The "Internet of Things," wearables, mobile and embedded technology, cloud, big data, robotics, and even holographic applications are where the action (and the jobs) will be. Get experience in one or more of these areas.

- Regardless of your view on the value of technical and professional certifications, hiring managers will continue to look for these qualifications as a starting point in selecting candidates.

- Hiring organizations, both public and private, expect new hires to hit the ground running, and there is little tolerance for an extended learning period.

- In some organizations, employers are investing less in employee development. This means we must proactively define and implement our skill development plans at our own cost and on our own time, if necessary.

- Professional networking remains a crucial element in our career journeys. Even if you're currently in an awesome job, you must not allow your professional network and industry contributions to the community go stale.

My overarching encouragement is to be constantly in touch with where our profession is headed and set a defined path for being prepared for your next role, whether inside your current organization or at a new company. This is your moment. Own it!

GONE ARE THE DAYS WHEN COMPANIES EMPLOY PEOPLE FOR A LIFETIME—OR WHEN EMPLOYEES WANT TO STAY WITH THE SAME COMPANY FOREVER.

Is Your Career Rewarding?

Often the best way to understand the present and consider the future is to look back on the past. I have the privilege of working with hundreds of developers and testers across many industries each year. One of the questions I'm always interested in is whether people are fulfilled in their roles. If so, why? If not, what are some ideas for moving toward greater satisfaction?

Titles are only business labels for a defined set of responsibilities and accountabilities. What really fulfills you in your career is far more complex than simply a job title. This presents a question: What are your life's goals that are important to you? Because most of us spend a third or more of our time at work, your career is a major aspect of overall fulfillment in your life.

The following things have been some of the most rewarding parts of my career.

- **Delivering a quality software product.** One that you and your team have worked hard to extract bugs from and, once deployed, the customer has provided positive feedback.

- **Teaching, mentoring, or coaching teams.** Passing on something I've learned to an individual or team and helping them to learn through exercises and examples. Helping teams work better together and assisting them in meeting their business objectives.

- **Leadership.** In a "servant leader" way—that is, providing guidance for others to solve challenges themselves, helping to remove roadblocks, jumping in to lend a hand, and collaborating on innovative ideas. This also includes building others up and offering them positive comments, feedback, and rewards.

- **Driving change.** Whether local or global change. Being part of a team that plans, evaluates, and then selects and implements a particular methodology or tool, for example.

- **Discovering breakthroughs.** Having the freedom to innovate. Working together with others in an entrepreneurial way, brainstorming, and being creative in solving problems or generating new ideas that offer more value to customers.

- **Developing global relationships.** During my career I've had the privilege of creating quality and testing groups worldwide and have been rewarded through some wonderful professional relationships.

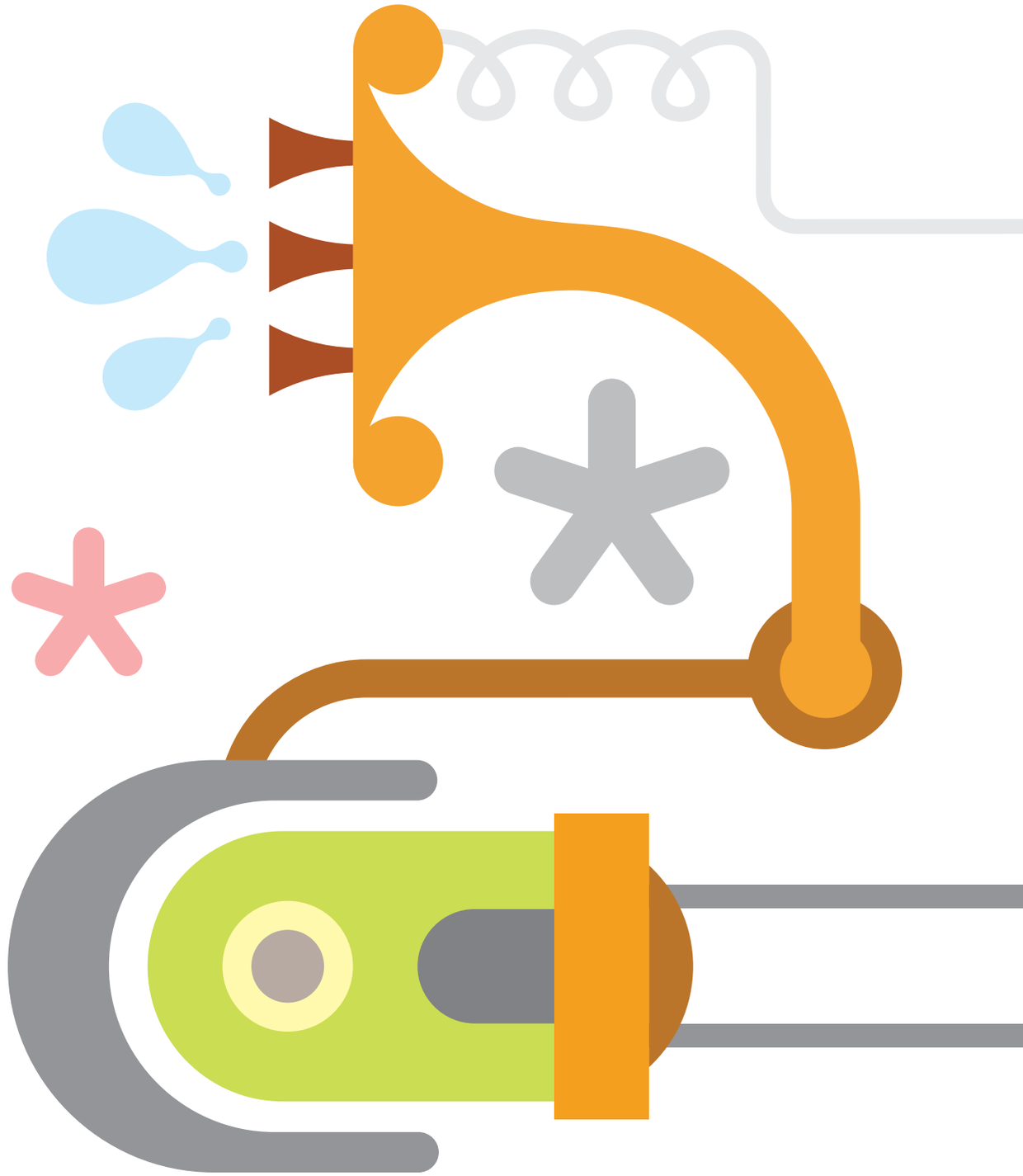
- **Contributing to our testing profession.** Speaking, collaborating, and authoring ideas to benefit our occupation. I wrote **a white paper on advancing the testing career** that addresses some of the challenges and strategies there.

If I were to net it out, what has been most fulfilling for me thus far in my career has been the professional relationships that have evolved into friendships, seeing others learn and grow, knowing that our team has made a positive impact on the customer, leaving an organization in a better place than I found it, and learning something new about myself along the way.

I am humbled each day by those who have contributed so much to the fulfillment of my career. Although there have certainly been many challenging situations (we call these learning opportunities) during my journey, my career as a tester, consultant, and business executive has been extremely rewarding.

Chapter 2

Testing in the Agile Age



Transitioning from a Traditional Tester to an Agile Tester

One of the most challenging changes any organization can make is moving from a traditional lifecycle development approach to an agile one. As such, there are many books, articles, blogs, and consulting firms that aim to assist with this transition.

One question I'm often asked is "What's the difference between a tester's role in a traditional lifecycle model versus in an agile methodology?" The answer is not an easy one. There are many factors to consider, and any individual tester's or test team's experience will vary based on the organizational environment, culture, and the technology being tested.

There is a spectrum of differences, ranging from redefining the testing role and responsibilities completely to making only minor changes in context and accountability. Some individual testers and test teams will find this transition easy because it's very close to the way they are already working; other individuals and teams will find the transition much more difficult, to the extent that they may need to reinvent themselves.

At the risk of oversimplifying, here are a few key differences in the context, responsibilities, behaviors, and expectations of an agile tester versus a traditional tester.

As an Agile Tester	As a Traditional Tester
Testing is conducted immediately and continually as soon as possible, with the smallest feature(s) available. Test-driven development is employed.	Testers usually wait on a specific build or release and then begin testing once most features are implemented.
Testing is planned as part of the sprint and the release. Developers automate unit tests. Functional and nonfunctional testing is conducted iteratively within the team and in collaboration with the product owner.	One phase of testing usually builds on the next—unit, then integration, then system, then acceptance.
Bug identification and repair is in hours rather than days or weeks.	There is significant wait time between bugs being identified and bugs getting fixed.
Developers and testers operate as one team and interact continuously and collaboratively. The testing voice is equally represented.	Testers are less a part of the development team. Testers may be more distant in interaction and communication with developers and may have less of a voice.
Testers and developers are part of a homogeneous team accountable for quality delivery of the system under test.	Testers are accountable for testing. Developers are accountable for developing.
Testing is continuous and all quality steps are planned and executed iteratively by the agile team.	While the goal is always to have quality built in at every step in the lifecycle, in practice, much of the checking (quality steps) occurs during the backend testing cycle.
Automation is a must have, particularly for unit tests, as it supports continuous integration.	Automation is not a necessity because most testing of new development is done manually.

An Agile Approach to Managing Your Software Testing Career

I recently had the privilege of speaking at STAR*EAST* 2015 on the **future of the testing profession** and whether software testing expertise is seeing its demise or rebirth. A link to the video of my presentation is included in Chapter 5 of this book.

To be clear, my view is that the need for software testing expertise has never been so critical. With the average consumer interacting with 250 computing devices per day, reliable software is not only a business responsibility; it's a societal requirement.

In my keynote, I went so far to say that as long as people are accountable for software development, there will be a need for skilled testing roles.

If you accept my assertion, how do you think we as testers should evaluate, plan, and manage our career journeys?

> THE NEED FOR SOFTWARE TESTING EXPERTISE HAS NEVER BEEN SO CRITICAL.

The first step is taking accountability for the growth of your expertise. I have written about owning your career already, so I will not repeat that information here. Rather, let's focus on an agile approach to determining your career direction, evaluating the alternatives, and developing a plan.

There are at least four dimensions to consider in evaluating and planning your career:

- **Industry passion:** The choices here are numerous: finance, insurance, banking, aerospace, healthcare, manufacturing, hospitality, education, transportation, retail…What area are you interested in?

- **Technology passion:** A cross section of industry studies will yield big data, cloud, mobile, embedded technology, wearables, security, analytics, the Internet of Things, service virtualization, and 3D printing as the hot technologies for the next decade. It would be smart to familiarize yourself with a few of these concentrations.

- **Role passion:** As organizations embrace agile, I see four broad roles for those with software testing expertise: test architect (a senior consultant or adviser role), coordinator of testing (a facilitator or project manager role), test automator (technical testing expert), and test designer (business testing expert).

- **Methodology passion:** Do you like more structure and a disciplined development approach or a more flexible and agile development environment? Does working on higher- or lower-risk products excite you? Are you more comfortable in an independent QA and testing team, or do you thrive on close, cross-functional team collaboration?

With these four dimensions, you can employ an agile approach to evaluating and planning your career.

In your first sprint (put half an hour a day for ten days on your calendar, as an example) you can select an industry area, a set of companies within that industry, a technology, a role, and a methodology passion to learn more about—for instance, a test architect in the healthcare industry focused on data security in a compliance environment, or a test automation expert in financial services focused on mobile in an agile environment. The minimum viable product at the end of each career sprint is the information to eliminate career alternatives and move forward to the next sprint with either other career choices or a deeper dive into a desired career path.

Regardless of the career planning approach used, the most crucial step is to just do it. Good luck sprinting your career journey!

Maintaining Tester Independence in an Agile World

As organizations adopt agile methodologies, one of the key challenges is reinventing traditional roles such as project manager, functional manager, business analyst, developer, and tester. While many of the tasks and accountabilities for these roles must still be carried out, they are set in a much different context: one of a homogenous agile product team.

Ideally, these responsibilities could be carried out by anyone on the agile product team, but in practice, each team member will have his or her own individual skills, experiences, and strengths. One primary shift is that the entire agile team is accountable for quality—carrying the quality flag is not the sole responsibility of the tester.

A key principle in testing is retaining the objective voice—that is, the "check" step in the **plan-do-check-act cycle** that W. Edwards Deming provided us. The goal is to have testing activities embedded in every step of the lifecycle and across all roles. There is value in having another set of eyes on the work of the originator, but we also want to ensure that we maintain tester role independence while assuring a highly collaborative, responsive, and flexible team environment.

ONE PRIMARY SHIFT IS THAT THE ENTIRE AGILE TEAM IS ACCOUNTABLE FOR QUALITY—CARRYING THE QUALITY FLAG IS NOT THE SOLE RESPONSIBILITY OF THE TESTER.

Organizations are approaching this in several ways; there's no one prescribed formula. Some keep their independent testing teams and then assign testers to each agile product team, and others abandon their functional testing departments and embed the testing within the agile teams. Still others do a little of both: create agile product teams that include testing roles and then also retain an independent testing team. And still others change reporting relationships in multiple ways.

Regardless, the people accountable for the testing role in an agile product team must be able to make an independent assessment of the system under test, and that requires the team members in this role to have both a mindset and a skill set in the art and science of testing.

I don't believe a best approach to maintaining tester role independence can be prescribed. Rather, each team and organization should tailor its strategy based on what it needs.

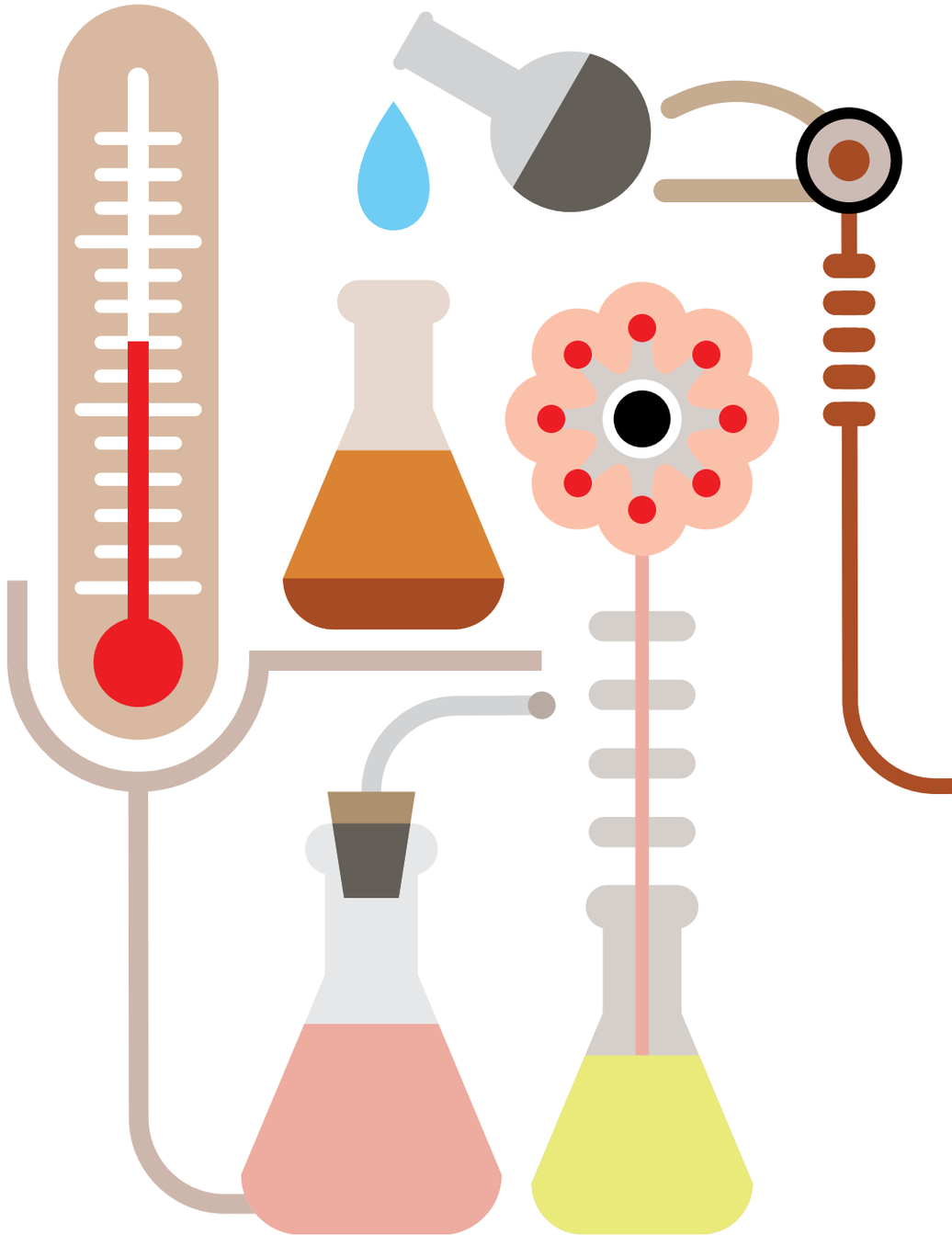
In the spirit of the Agile Manifesto, here are my "work in progress" values to consider in maintaining tester independence in the context of an agile product team.

We value:

- Product team accountability for the right level of quality delivered at the right time
- Product team collaboration while embracing the best skill sets of each contributor
- Planning for quality and testing at every step of the release and within each sprint
- Checking at every step by the originator and by one or more independent assessors

EACH TEAM AND ORGANIZATION SHOULD TAILOR ITS STRATEGY BASED ON WHAT IT NEEDS.

Chapter 3
Tools and Techniques



More Slack Time, Please!

I'm all about getting things done. This means that daily, even hourly, I'm putting things on my to-do list, checking them off, delegating actions, following up on delegated actions, moving projects forward, starting new projects, maintaining existing projects, running to meetings, accepting new actions from those meetings, planning sprints, attending standups, completing story testing—all while responding to email, chat threads, texts, phone calls, and voicemails. Like most of us, my days and weeks easily and quickly fill up with tasks.

While delivering results is expected (and what we get paid for), there is a consequence to this fast-paced, results-only focus. In short, it can severely limit our ability to think, to grow, and to change. In his book ***Slack: Getting Past Burnout, Busywork, and the Myth of Total Efficiency,*** Tom DeMarco captures this behavior nicely:

> We live in an age of acceleration. Whatever the formula was for business success a few years ago, it won't work today. Today there needs to be more and more work crammed into less and less time. There are few people doing more and doing it faster in less space with less support and with tighter tolerances and higher quality requirements than ever before. There is no time for analysis, invention, training, strategic thinking, contemplation or lunch.

As we seek and achieve efficiency, we eliminate “slack time”; after all, slack is waste, right? If I’m not producing tangible output, then I must be on a worthless “idle” mode, correct?

No, absolutely not! By “slack” I mean purposeful time to allow our brains individually and our organizations collectively to create, think, reflect, analyze, contemplate, plan, learn, grow, and change. It’s the downtime required to allow you or your organization to breathe.

SLACK IS THE FUEL THAT DRIVES OUR ABILITIES TO INNOVATE, EVOLVE, AND CHANGE FOR THE BENEFIT OF OURSELVES AND OUR CUSTOMERS.

Here are some ideas for building more slack time into your own or your company’s routines:

- Take a look at those sprints. Are they continually back to back with no time in between for team learning or process or technology improvements?
- Be intentional by putting time on your calendar for research, reading, and attending learning events such as courses, conferences, or web seminars.
- Determine when and how you are most creative. What environment? What time of day? Then purposefully set aside some quiet time in those ideal conditions.
- Ask people in your organization how many times a week they feel like they have tasks scheduled back to back. Brainstorm together about whether any meetings can be consolidated or eliminated, and think about organizing some collective company slack time.
- Evaluate the amount of time you and your team are spending delivering results versus thinking about the next innovation. Allocate some time for creativity to encourage improvements.

All of the above suggestions are obvious. The challenge is having the discipline to act and the courage to intentionally create slack.

Slack is the fuel that drives our abilities to innovate, evolve, and change for the benefit of ourselves and our customers. Start slacking today!

The Value of Checklists in Our Software Testing Work

How do you keep track of all the details in you busy professional and personal life? Do you have a to-do list, write things on napkins or the palm of your hand, or just keep items organized in your great memory? I’ve always been a to-do list kind of a guy, either using a paper list or, more recently, an electronic list of my tasks.

There's another kind of a list that many of us (myself included) may dismiss more quickly than we should: the checklist. By checklist I mean some list that details a specific set of items, normally in an organized sequence, to ensure you have not forgotten any one in a series of crucial steps.

Some may see checklists as unnecessary, but consider the growing complexity of many of our software development and testing tasks. While in some cases testing tasks may be much less risky than shooting a rocket into space, flying an airplane, or conducting brain surgery, in other cases of life-or-death and mission-critical software, these are a must.

Imagine SpaceX lifting off its next rocket—or a surgeon operating on you—without a comprehensive checklist. Closer to home, think of the cost in missing a step in your complex software testing work. At a minimum there will be rework, or worse, a critical bug that will impact your customers. As testing professionals, the complexity of our responsibilities continues to increase exponentially with each new technology that is integrated into our systems.

The following are some examples of checklists:

- ✔ **Test design checklist:** Have all the steps in carrying out comprehensive test design been implemented?
- ✔ **Test plan checklist:** A detailed test plan template that itemizes and describes the content that is expected in each test plan section
- ✔ **Test case checklist:** What are all the dimensions of a test case that must be described—preconditions, input, expected results, postconditions, and so forth?
- ✔ **Nonfunctional checklist:** A performance test checklist. What are all of the detailed items or steps to be executed (and in what order) for load and stress testing?
- ✔ **Agile sprint testing checklist:** What are all the testing tasks (functional, nonfunctional, and exploratory) that need to be done for each sprint?

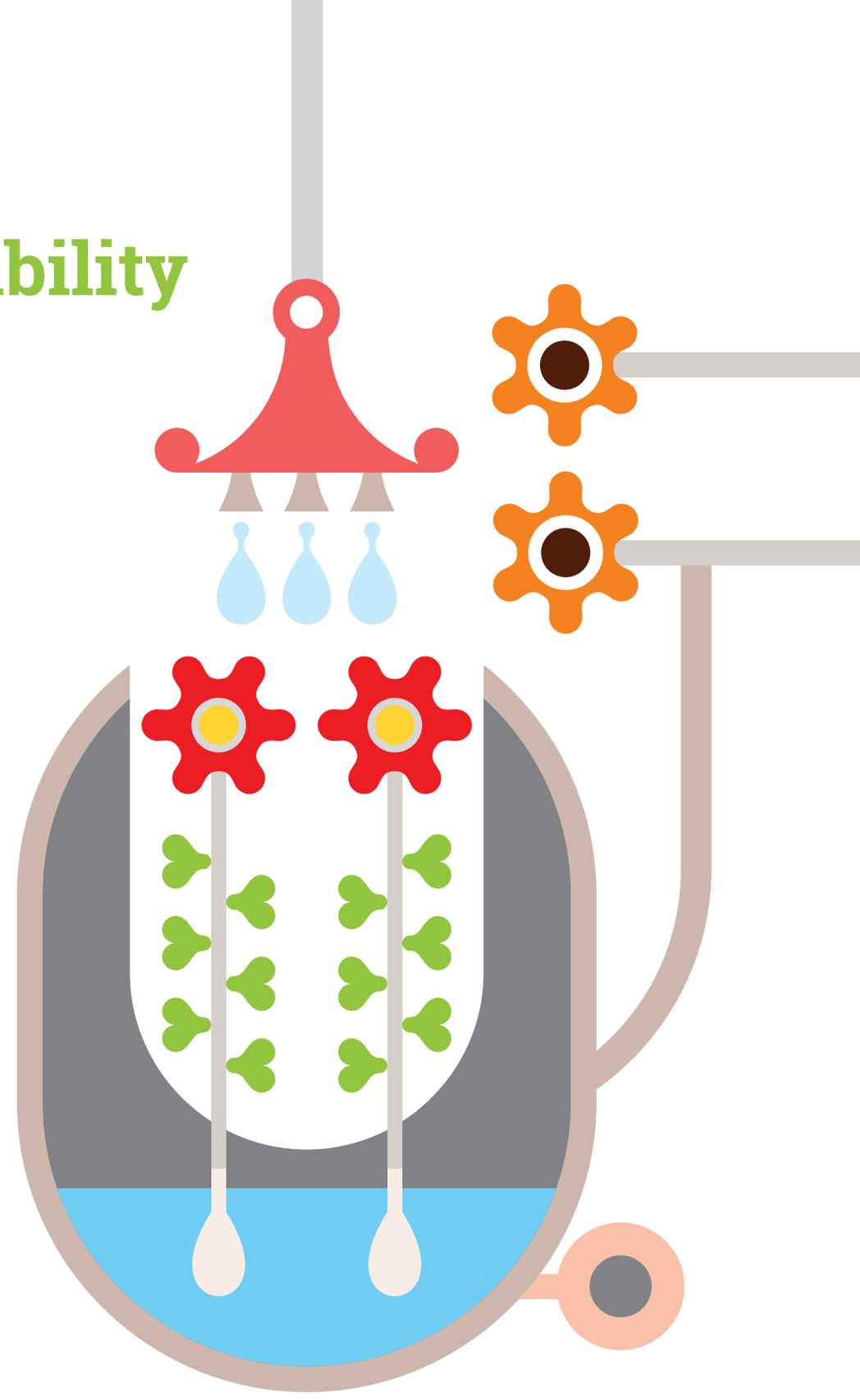
In his book ***The Checklist Manifesto,*** Atul Gawande makes a distinction between errors of ignorance (mistakes we make because we don't know enough) and errors of ineptitude (mistakes we make because we don't make proper use of what we know). Failure in the modern world, he writes, is really about the second of these errors.

IN CASES OF LIFE-OR-DEATH AND MISSION-CRITICAL SOFTWARE, CHECKLISTS ARE A MUST.

Making mistakes because we didn't make proper use of what we know is simply bad professional practice. Ignoring the use of a tool such as checklists, where appropriate and it would add value, increases the risk in our already risky world of software testing.

Chapter 4

Leadership and Social Responsibility



Be Big, Bold, and Brave in Your Testing Efforts

I've worked for some wonderful mentors and coaches during my career. One of those was the business president I worked for at a leading financial institution. She was an awesome leader and highly energized about motivating her team to achieve our stated business goals. She held several tenets, one of which was "Be big, be bold, be brave"—meaning don't be complacent, take reasonable risks, and don't be afraid to try something innovative.

Our organizations, management, teams, and customers desperately need each of us to step up and lead. Regardless of whether you have an official title as a leader or you are an individual contributor, you must exercise leadership in your role.

Early in my career I thought having a title would allow me to influence people in my organization. If only they'd give me that promotion to manager, then people would listen to me; after all, I would be "knighted" to lead.

I quickly learned that titles make little difference in the way we are perceived. What counts are our abilities to contribute to the organization by delivering results, solving difficult challenges using our creativity and innovation, having a set of core values that are lived out each day, offering professional insights and new ideas, and communicating a vision in the context of the subject under discussion.

I recently had minor knee surgery. During the course of working with my doctor, nurses, and other technicians, three technology issues arose. On the first visit to my doctor, his office had converted to using a fully digital means of running their business, keeping all medical records up to date using wireless laptops that were wheeled around between patient rooms. The day I visited, however, the technology was not working, so no records could be accessed.

A few weeks after that, I arrived at the hospital again and none of the nurses could log into their roaming desktop accounts. The system said they were already logged in. The third issue arose when I received a bill for an MRI that the insurance had already agreed to cover. An inquiry about how this happened pointed to a billing software error.

None of these issues directly impacted my surgery, but I observed a high sense of frustration among the users and wondered what indirect effect this "hassle factor" might have on the attitudes of those who were there to serve patients rather than waste time navigating through technology hurdles. Worse yet, no one was taking the lead or acting accountable for resolving the issues I observed.

These situations reinforced for me the critical nature of our roles as testers and the need for each of us to be big, bold, and brave. Are you taking the initiative in your testing projects to try to ensure that issues like those I observed are being avoided?

Software Testing: A Social Responsibility

In the late '90s I authored an article about the accelerated pace of software development driven by better development languages, tools, and reusable code, and the fact that we were producing more software than we could test. While this may be just as true today as it was more than a decade ago, there's another critical dimension contributing to the software risk equation: the ubiquitous deployment of software.

THERE ARE FEW WAKING HOURS NOT HIGHLY DEPENDENT ON TECHNOLOGY WORKING CORRECTLY AND RELIABLY.

Think about it: How many computers, running one or more software programs, have you interacted with today? There are few waking hours not highly dependent on technology working correctly and reliably. Marc Andreessen famously **wrote in *The Wall Street Journal*** in 2011, "More and more major businesses and industries are being run on software and delivered as online services—from movies to agriculture to national defense."

This trend will continue to accelerate as we embrace more cloud capabilities and move toward the Internet of Things, where information and software capabilities will be more "push" (the information automatically shows up and the tasks get automatically done) than "pull" (we invoked apps to do those actions).

As businesses and consumers embrace big data and analytics, mobile, cloud, the IoT, social media, and many other rapidly emerging technologies, the expectation that "it just works" is rising exponentially. Equipping our existing technical workforce with the capabilities to build and evaluate software is crucial to protecting our software-driven global economy.

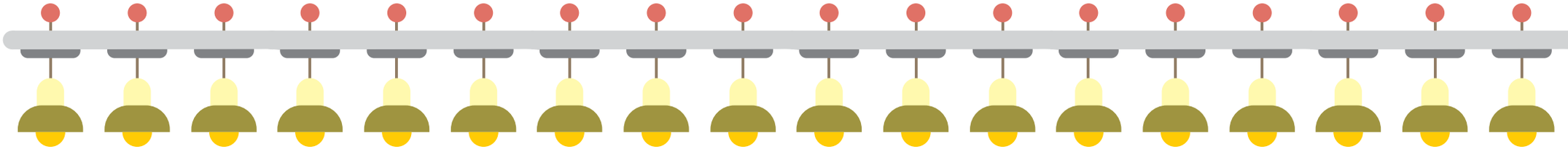
Consequently, those who specify, develop, test, and deliver software have a tremendous accountability, and they must be supported and equipped with the best education and training, tools, and approaches. Further, there is tremendous opportunity for those considering roles as developers and testers to acquire these skills and contribute to the continued health of software. We need more experts to mitigate the risk in our software, and there are a plethora of opportunities.

The employment search engine Indeed listed **software quality assurance engineers and testers** as one of the top ten jobs in demand for 2015. Interested, qualified people could become part of a company that needs their expertise, join a professional software services firm, jump on the crowdsourcing bandwagon as a freelancer, work with a software tools or methods supplier, or other possibilities.

WE NEED MORE EXPERTS TO MITIGATE THE RISK IN OUR SOFTWARE, AND THERE ARE A PLETHORA OF OPPORTUNITIES.

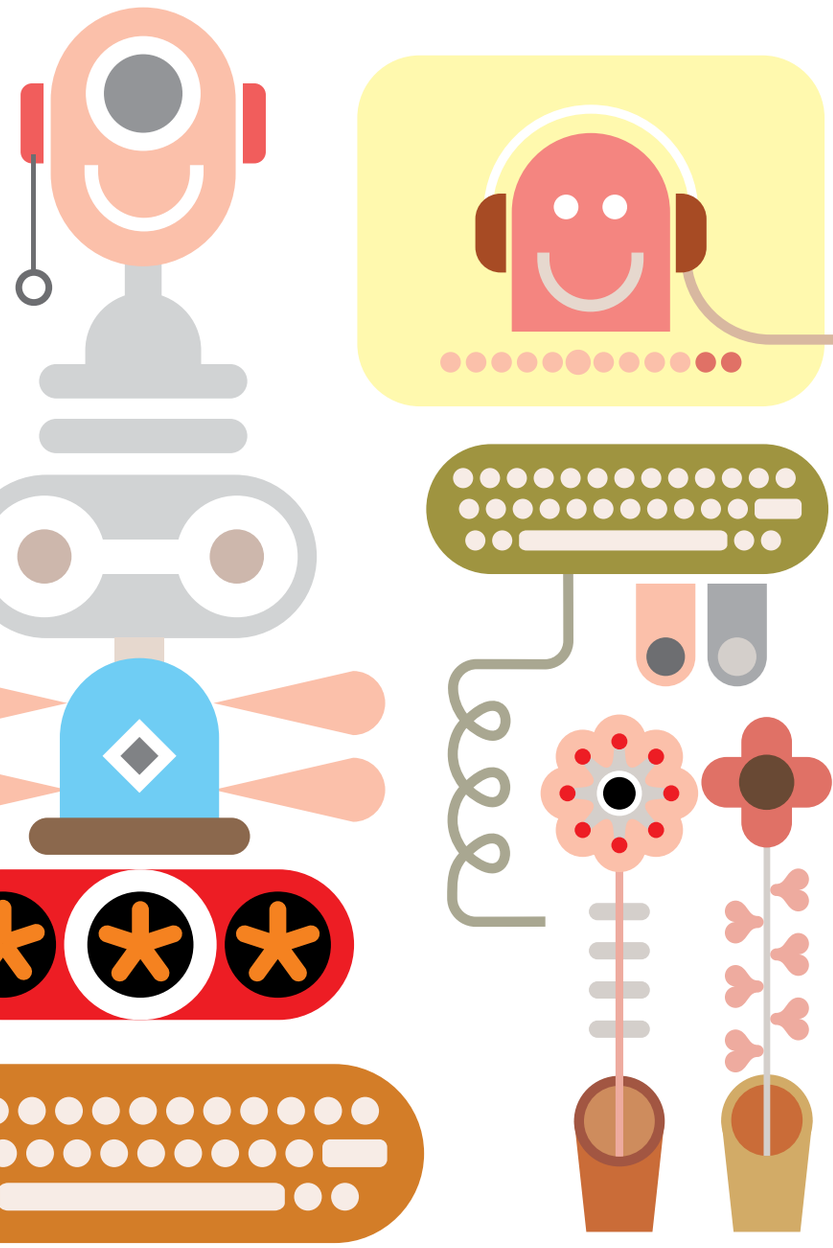
If you're already a member of our critical software economy workforce, keep learning. And if you're considering a software career, now is the time.

Software is now a social responsibility. The demand has never been greater, the opportunity has never been better, and the reason for action has never been more compelling.



Chapter 5

The Future of the Testing Profession



Five Predictions: What the Future Holds for Test Professionals

No one has a crystal ball, but it's critical for test professionals to seriously ponder what the future holds as they plan their career journeys. Where is technology headed? What key software development methods will be embraced? Which tool is likely to be the mainstay? What testing roles will be in most demand? Which will be most valued? What tester role will be best compensated? Where will a tester spend the most time? What technical skills, certifications, or accreditations will testers need? Here are some predictions based on a few of those questions.

1. **Technology:** Cloud, mobile, embedded software, and wearables are mentioned in just about every industry article right now. Testers should absorb as much as they can about these technologies.

2. **Methods:** There will always be a place for waterfall, but agile is now the norm. The challenge, it seems, are the many forms of agile methods, both formally defined and homegrown. Nonetheless, having experience in an agile environment is a requirement.

3. **Tools:** All signs point to becoming more technical and acquiring competency in one or more programming languages. Of course, this does not obviate the need for testers who are business experts.

4. **Testing roles:** Three "A" words come to mind: architect, analyst, and automator. As more organizations recognize the need and value of a test architect, this role will have the opportunity to better influence product design and quality (defect prevention). Test analysts will play an increasingly important role in designing effective and efficient tests (defect containment), and test automators will be in demand given the need to run comprehensive test suites quickly.

5. **Skills and qualifications:** Specialization in a given technology, domain, industry, and role will become increasingly valued over a "test generalist." For example, a test architect in a health-care company working on security and privacy for mobile applications would be sought after over someone who identifies as simply a software tester.

It's been said that the best way to predict the future is to invent it. Test professionals are confronted with challenges in performing their work effectively and efficiently while improving their work and profession. In doing so, they must invest in their futures with well-defined career plans based on their predictions for the industry.

Video: The Future of the Software Testing Profession

The world of testers and test managers—like most professions—continues to evolve. Some say the more things change, the more things stay the same; others say that testing as a profession is dying. These divergent views raise compelling questions. Are we approaching the era of minimal defects in which testing is diminished? Or is testing on the brink of becoming the most important aspect of software development as the risk of failure grows exponentially? What role will testers play on development teams? What critical skills will testers need in the future?

In this conference video from STAR*EAST* 2015, Mike Sowers presents his and others' views of the key drivers that are shaping the future role of software testers and test leaders. Mike explores how testing is impacted by technology (cloud, mobile, wearables), process (development and testing methodologies), and innovation. He then shares observations on and recommendations for staying competent, competitive, and relevant as a results-driven dev/test team member in your organization.

About the Author

Michael Sowers has more than twenty-five years of practical experience as a global quality and test leader across multiple industries. He has led internationally distributed quality and test teams and held accountability for configuration management and release engineering functions. Michael is a senior consultant skilled in working with organizations, both large and small, to improve their software development, testing, and delivery approaches. He has worked with companies such as Fidelity Investments, CA, PepsiCo, FedEx, Southwest Airlines, Wells Fargo, ADP, Lockheed, and WellPoint Health Networks to improve software quality, reduce time to market, and decrease costs. Michael has mentored and coached senior software leaders, small teams, and direct contributors worldwide and has a passion for helping teams deliver software "faster, better, and cheaper."