Career Paths in Enterprise System Testing

Authors:
Tara Astigarraga¹, Michael Browne¹, Eli M. Dow¹, Christina Lara¹, Michael S. Phay²,
Dorian Thomas², Maria Ward¹, Nicholas Wiggins²

IBM¹; University of Maryland Eastern Shore²

Abstract

In order to produce quality enterprise products, companies employ hardware and software testers to go beyond test design and execution and engage in problem solving, debugging, and analysis in complex computing environments. Many graduates enter the work force with exceptional development skills, but lack proficiency in test design, test execution, debug, and analysis skills. While many universities focus on the development side of software engineering, few provide the same rigorous attention to the software testing discipline. This paper presents an outline of enterprise testing career paths from various industry employers. We believe the elucidation of the enterprise test career path will encourage universities to adjust curricula in order to produce high-quality system testers and system thinkers. In addition, we propose that companies seeking to establish dedicated test career paths do so by making use of the emerging T-shaped maturity model for growing and fostering test engineering skills.

1. Enterprise Computing and Systems Testing

As the enterprise computing technologies and customer demand continue to grow in complexity and scope, it becomes imperative that rigorous and methodical system testing keeps pace. Enterprise computing refers to mission-critical hardware and software upon which a business depends. Example industry segments which use enterprise computing platforms include: banking, retail, and health care. Errors or product failures in the enterprise computing systems which support these fields can lead to costly, catastrophic or in certain cases, life threatening events.

The complexity found in enterprise computing environments is astounding. Most customer environments are heterogeneous mixes of multiple vendor solutions. It is not uncommon for the simplest business process transaction to span multiple types of server hardware, I/O adapters, I/O switching devices, storage subsystems and network switching/connectivity devices. Furthermore, that transaction might require a different vendor for each component. This scenario only exposes the complexity of the hardware systems in place. When considering firmware, operating systems, user-level software, virtualization options, applications, and various software configuration parameters, the complexity increases exponentially. These complex configurations are the reality. In fact, the IBM System Storage Interoperability Center (SSIC) cites over 171 million supported configurations [SSIC].

As complexity continues to rise, so does the potential for errors. According to a 2002 study commissioned by the US Department of Commerce’ National Institute of Standards and Technology, the annual cost of software bugs to the U.S. economy is estimated to be $59.5 billion [NIST]. For example, the 1997 crash of Korean Airlines 747 in Guam resulted in 200 deaths was attributed to a faulty software configuration [NTSB]. Another example includes nearly 30,000 deaths and 600,000 injuries attributed to medical device software failures reported...
between 1985 and 2005 [CDRH]. A third example which emphasizes the importance of enterprise systems testing occurred when defective anti-lock brakes forced the recall of 39,000 trucks and tractors and 6,000 school buses in 2000 [CBS]. The problem was demonstrated yet again when the $165 million dollar Mars Polar Lander Space Probe was destroyed in its final descent in 1999, likely because its software shut the engines off 100 feet above the surface [CBS]. Additional examples are not difficult to find.

It is estimated that a bug found in a production customer environment can cost 150 times those found during early requirements reviews [SUPER]. The earlier hardware or software bugs are found, the smaller the cost associated to fixing them. This rationale stems from the increasing cost of late changes. These changes incur ripple effects expending additional unplanned resources through re-engineering code, documentation, redesign of hardware or software to provide the required fix.

Furthermore, if a product defect is found by a customer, as opposed to a tester working in a development or test lab, serious and expensive consequences may occur. Examples include customer support costs, loss of future sales due to decreased customer satisfaction or poor product reputation in the marketplace. Due to the elevated cost, customer impact, and defect visibility of later life-cycle fixes, companies are taking increasingly pro-active measures in order to detect and fix bugs (hardware or software) during the earliest stages of development.

In any industry where quality is expected, and defects can cause costly or even life-threatening errors, it is imperative that the shortage of software test skills among recent university graduates is addressed. In an age of increased attention to responsibility, both corporate and personal, software engineering must be held to the same rigor as other engineering fields, such as civil or mechanical engineering. As such, more effort needs to be placed on rigorous testing. More formal resources and professional support must be made available for engineers choosing to follow a career in systems testing if they are to grow and excel in an increasingly difficult and challenging field.

This paper will further describe the characteristics required of a successful systems tester and will explain the different types of knowledge profiles that are common among systems testing professionals. These examples will be further used to describe the effects on careers in enterprise computing.

**Systems Thinking Considerations**

Systems thinking is a way of looking at the world and understanding its components. Systems thinking has traditionally been applied to concepts such as economics, politics, and business, yet is also extremely relevant in the realm of software and hardware testing of computer or technology-based infrastructures. This is because each computer, with its collection of discrete hardware components and software can be viewed as a system. Even more importantly collections of computers form larger systems, yet the rules of systems thinking must still apply. Though books have been written regarding the abstract language of systems thinking as applied to non-computer domains, there is little published work with respect to computing. IBM has recognized a need for systems thinkers in computing. Others in the computing industry, such as Oracle (formerly Sun Microsystems) also have top technical talent claiming that there is a distinct lack of systems thinkers who understand the complex interactions between software and hardware in the real world [BMC].

To become a more effective programmer, system administrator, or software tester, one must understand, at some level, everything going on in the system. Each interaction can cause
ripple effects, unintended consequences, or problems. Consultants or systems programmers are often called upon to debug large systems without having complete knowledge of the system. This requires the ability to build mental models of what is going on in the system. Unfortunately, the ability to do this varies greatly among those called upon to help analyze and debug problems. This quickly becomes evident in the field.

Likewise, having the proper knowledge of a system enables exploiting operational or technical efficiencies and synergies. But how can someone who has only attended one or perhaps even a couple classes of enterprise systems learn to more quickly grasp new and different systems? The answer is by learning from those who have seen systems much larger and more complex. The enterprise systems tester, as well as systems programmers, must be trained to recognize patterns and anti-patterns that appear across all systems, so that they can operate on them appropriately. In essence they must learn the methodology of systems thinking to bring their skills to a higher level.

**The I & T shaped model and the impact on Career Potential**

Those who have an I-shaped knowledge profile have a singular, deep, foundation of expertise. They are usually uncomfortable when branching into other areas due to their lack of skill diversity. An individual with these qualities may find initial success because of their expertise in a given field, yet they will often find their ceiling for advancement to be very limited. I-shaped knowledge profiles are most commonly created in a traditional university education. Selecting a single major, and mastering that topic, creates I-shaped thinkers.

An individual with a T-shaped knowledge profile has a deep foundation in one area of expertise, but accompanies this with knowledge and understanding of other related areas of study and work experience. Such a person looks to diversify their overall knowledge within their chosen field of study. Typically these individuals choose a major such as Computer Science and accompany this foundation with added skills such as testing, business, or technical writing. Their expanded repertoire of experience enables them to more effectively contribute to their company.

This should not be confused with taking required liberal arts courses alongside a single technical domain. While liberal arts skills are absolutely vital to enable enterprise system testers to communicate at different levels, venues, and cope with organizational challenges critical to success, it is the lack of technical breadth coupled with the lack of formal systems thinking education that reduces the number of T-shaped enterprise systems testers in industry today.

**Qualities of exemplary system testers**

Certain attributes have been found to be inherent, core skills of exemplary test engineers that demonstrate the T-shaped knowledge model along with systems thinking skills.

First and foremost is the inherent ability to find defects. This is the ability to be able to look across a system or across various components to find defects, beyond the obvious broken functions, failed test cases, or extreme performance issues. One needs to be able to see nuanced behaviors and determine how to provoke undesired behavior. The need for systems thinking is evident at this level of enterprise systems test, but other skills may not be so obvious. Having a robust set of performance analysis skills, especially at the cross system level, are a must for success. The ability to find defects at multiple points in the product life cycle is extremely important. A test engineer must be able to find defects at design time, in the lab environment, and in the customer environment.
The understanding and use of automation techniques is also extremely useful to the enterprise test engineer. Test engineers should understand how to make use of the automation tools in their inventory, yet must be aware of the dangers of over-reliance on tools. There are known anti-patterns associated with over-reliance on automation which can incorrectly lead testers to believe that a 100% automated test success rate translates to a 100% working product or service[AOT].

Design knowledge of the product or service being tested is also a critical requirement to properly test an enterprise product or service. This knowledge also has to be combined with customer usage knowledge as well. These areas draw from the horizontal component of a T-shaped thinker.

Lastly, a test engineer must have superior test design skills that combine with the skills mentioned previously, to enable the efficient design of both the tests and the environment used to perform the tests. This efficiency of design is required to demonstrate that the product or service will not break. In addition, the enterprise test engineer must strive to reduce the time and cost of the test process. This activity is often called “cheating” at testing and is often an area of innovation and invention for test engineers.

2. Careers in Enterprise System Testing

Based on a survey of various Internet articles, the authors’ accumulated experience in the industry, on-line salary surveys, and data gleaned from various technical job posting sites available at the time of this writing, we have concluded that the majority of test engineering career paths generally fall into three major categories: Tester to Developer, Tester to Quality Assurance (QA) manager/director and Tester to Test Engineer and Technical Executive. We feel the data has provided a fairly consistent pattern. Note that these paths are not mutually exclusive, nor absolute. One can move from a given path to another during the course of a professional testing career, although a singular path is most often chosen and followed by the most successful individuals.

**Tester to Developer**

The path from tester to developer is a very typical path for newly hired developers, or those who enter product or service development careers through a non-traditional training background. This person is usually placed in a development team and given function test assignments. A person on this path is typically an I-shaped person, or a person who is targeting the maximum career achievement attained with I-shaped learning.

The individual usually will work in the test environment for a few product releases and may then move to different components within their existing product or service development team. They will often gain a promotion or two during the course of this role and further move on to take technical leadership of a small development component. Sometimes they might even continue to test adjacent components. As the individual acquires more skill in development, they often transition into roles where they are responsible for larger, and more complex, components within the product solution and relinquish testing responsibilities altogether.

With agile development processes becoming the normal approach to development, this type of career path will become very typical as it exposes the would-be-developer to the code base and instills the value of testing which dovetails with test-driven development practices. If
no move to development is made, the individual will often move to a more formalized test role within their current organization or to a formal test organization if one exists.

**Tester to QA Manager/Director**

In this career path, an individual usually begins their test engineering career with the intent of remaining a tester throughout their career. In other cases, they may have been in another development oriented career path while discovering that they prefer testing activities or management aspects over traditional development activities. It should be noted that an increase in the availability of test curriculum at the university undergraduate level will allow students to make this type of career choice when selecting a place of employment.

A typical path for the test engineer would be to move across components within a product as well as moving across test phases of a component. Typically, they will gravitate to later test phases such as systems test or integration test. These latter test phases are where T-shaped experience and thinking are critically important. In addition, the range of products and components under test can be quite wide, thus providing opportunity to gain breadth for the adventurous. This testing career path is quite common in enterprise class products as the complexity of those environments creates a strong demand for test engineers, and thus provides the justification for a formal test engineering career path.

As one progresses down this path, they will typically do normal test engineering activities that span: test execution, test development, test design, product debug and data capture. After achieving a broad and deep technical knowledge base, and quite likely a few promotions as a result of their work, they will usually encounter a fork in their career path. They can either opt to continue their career as a test project manager, or they may elect to pursue a test architect position.

The project management role is typically based around managing the test execution while a test architect focuses on the planning and design of tests for future product releases. The test architect will also act as technical lead for decisions regarding known defects. Both roles, architect and project manager, have career growth opportunities resulting in senior engineer levels in product development organizations. Typical titles for top achievement levels are “Senior Quality Assurance Analyst”, “Senior Test Engineer” and “Enterprise Test Engineer” [ATT]. It should be noted that although these individuals are somewhat T-Shaped, they retain a test-centric view of the world and take input from development or service analysts. Their customer interactions are typically second hand and usually involve recreating customer located defects in the product development labs. While they require expert level skills in both test and product domains, this person’s career usually plateaus as this level.

To further advance, a test engineer will often enter a management role responsible in some part for both the personnel and business sides of the testing discipline. In a large organization there is often a director or vice president of quality assurance which are normally executive management level positions. It should be noted that in many cases, executive management positions in test organizations may be filled with traditional product development executives seeking to broaden their experience, through exposure to test activities and management, as opposed to filling the role with an organically grown quality assurance director or vice president.
Tester to Test Engineer and Technical Executive

In this final enterprise test career path, the tester often starts out as a test engineer and continues on to become a test architect. The difference from the previously discussed career path is the level of domain knowledge depth is often quite deep by comparison, and the individual will often focus on debugging defects down to the level of determining offending lines of code. Most times, they will become very involved with their development team and will actively participate in the design of the product. It is quite common for them to be very involved with the service team resolving customer issues. A career-long tester in this role will also possess skills at various depths in many cross system disciplines. They most often foster significant customer relationships, which in turn make them a more educated test engineer. Industry standards and academic involvement are also expected and commonplace. When this level of depth and breadth is obtained, the tester will relinquish the role of test architect usually after only a couple releases in that capacity. This person moves to an extreme T-shaped model, fulfilling similar roles at multiple levels across the company and within the industry.

Technical influence regarding how, and which items, should be tested becomes critical. These test experts are often in such demand that participating in only the normal activities of test architect for a single product or release would dramatically decrease their ultimate value to the organization as a whole. This role is vastly different than managing a test cycle. Usually an individual on this path has no interest in management activities which support a test cycle. They are often called upon to make purely technical decisions about whether or not to ship a product based on their deep understanding of actual customer environments and the technologies under test.

In order to maintain this level of technical expertise, these individuals almost always return to hands-on-testing of products during the most complex test scenarios. They will often work with hands on test engineers more often than their counterparts who pursued management roles, sometimes performing actual tests on a day-to-day basis. This return to hands-on-testing is a critical vehicle for improving an organization’s continuous improvement culture. When a T-Shaped person with advanced depth and breadth returns to the hands-on environment an intended side effect is that the technical expectations of other test engineers is increased. Having a small number of individuals capable to fulfill this role is an essential part of enterprise testing. It is also essential for advancing both the test discipline and the product being tested. Ultimately, one would like to have a larger number of this level of test engineer, but T-Shaped people with this level of technical, as well as business, breadth and depth are uncommon.

A test engineer in this role is usually a Senior Technical Staff Member or a Distinguished Engineer. Historically the software testing discipline did not have individuals at this level. The enterprise complexity of an IT environment today is driving the test discipline to embrace the cultivation of this level of test engineer. IBM's Systems and Technology Group have had test engineers at these levels for over a decade. Other companies have also created comparable positions at these same levels. For example, Juniper Networks (as of the writing of this paper) are advertising for a System Test Distinguished Engineer [JNPR]. The job description requires deep hands-on experience in, programming, customer deployment, industry involvement, invention/innovation, as well as expert testing. This level of expertise is strongly comparable to the distinguished engineers found in traditional development roles.
3. Enterprise System Testing – From Pre-Hire to Retire

James Whittaker of Google, provides some insight into the need for testers to have a degree of technical sophistication in his oft-cited work [WHIT], "To plan and execute tests, software testers must consider the software and the function it computes, the inputs and how they can be combined, and the environment in which the software will eventually operate. This difficult, time-consuming process requires technical sophistication and proper planning. Testers must not only have good development skills—testing often requires a great deal of coding—but also be knowledgeable in formal languages, graph theory, and algorithms. Indeed, creative testers have brought many related computing disciplines to bear on testing problems, often with impressive results."

This section of the paper describes an overview of the steps required for high achievement in the test engineering disciplines beginning from college student, through the career paths of early and late career. We begin with the discussion of collegiate activities.

As a Computer Science Major

Computer science students looking to achieve a high rate of success in enterprise software testing should certainly have some math background to augment their programming skills, such as a general course on statistics or probability. In addition to the technical aspects of a computer science degree, the motivated student should also strive to take courses in technical writing when possible. Enterprise system testers are responsible for writing technical documentation, user messages for the software under test, patent documents to describe test inventions, as well as publishing written works which are often accompanied by a public speaking engagement.

For the core technical course, in computer science or software engineering one should strive to have both a theoretical and practical background. General programming courses which span a variety of languages will prepare the student for whatever future code base they will be testing. In addition, the wise student will seek out heterogeneous operating systems exposure. Those who know Microsoft Windows as well as other operating systems like Linux will find it easier to test other enterprise operating systems such as z/OS, z/VM, AIX, HP-UX, and Oracle Solaris when compared to those who have a homogeneous computing experience.

Nearly all computer science curricula provide courses related to software development, however the number of courses related to software test is limited at best. Students taking dedicated testing courses or courses with strong test curricula gain key systems test foundational principles. The knowledge gained from dedicated test classes can also help to broaden the student’s career path options. If no dedicated testing course is offered, asking a professor for a directed study in software testing may make sense. In addition, it is highly recommended that a student take any formal verification methods courses offered. Students may also opt to attend academic conferences on software testing, test engineering or even software development practices.

It is important that the well rounded computer science student study software development models (agile and waterfall) in order to be prepared for whatever development method is used in the corporate culture you eventually become a part of. Additionally, the student should engage in team-based software development practices and competitions to create large scale software that someone might actually use. We recommend that students work on a
large body of code at least once, for instance adding a feature or bug fix to an existing open-source code base boosts practical experience. It is likely that doing so will provide a practical introduction to one or more bug tracking software tools while concretely demonstrating knowledge of software defect life-cycles and defect tracking.

Along these same lines, learning a variety of software development tools, IDEs, and debuggers will make it easier to learn additional tools while once the job comes. Once more, broad exposure helps to create a well rounded tester.

Corporations within the IT Industry are always seeking students with background, knowledge and practical skills related to systems test. Students with this skill-set will distinguish themselves from their peers and position themselves as a valuable asset to any company.

As a Finance Major

As a finance student looking to begin their career in Enterprise Computing, it is very important to begin adding technical breadth to your core understanding of finance as soon as possible. Beginning with the freshman year, students should seek out opportunities to diversify their skills. Introductory courses allow students to explore other technical majors and allow the time needed for those students to determine their technical passion. Freshman should elect courses which create a strong foundation for core business knowledge.

Sophomore business students should focus on narrowing their major to an area of personal interest. Sophomore business students, like those in computer science, should embrace classes such as Technical Writing and English, as they allow the student to build, or refine, their communication skills. Sophomores have the opportunity to explore different minors which contribute to the technical and or business breadth needed for a T-shaped knowledge profile.

The first semester of the junior year should be spent solidifying the depth in your major, by focusing one's attention on creating expertise in their given field. The latter semester of a junior year provides the student with additional time to work on depth, or if already achieving sufficient depth, to branch out in search of additional breadth. We challenge business students to reach out from their comfort zone and to take classes that instill critical thinking skills as an undergraduate. Classes such as programming or courses dedicated in enterprise computing would be recommended for the finance students focused on a career in enterprise systems testing or careers that require this type of work like IT technical marketing. For the financial industry where doing things like proof of concept deployments is important, developing technical marketing material like IBM Redbooks often include many elements of system testing. These courses allow a graduate to pursue working in the financial services domain as well as enabling products and offerings in the financial industry.

As an Accounting Major

The addition of computer science courses is a perfect example of branching out from an I-shaped student to the T-shaped model. With an accounting background, knowledge in Enterprise Systems is extremely beneficial. This is an excellent choice for supplementing the basic I-structured accounting curriculum. The inclusion of computer science courses, particularly those geared towards enterprise systems, offers an introduction to unfamiliar subject matter. For any student seeking a T-shaped knowledge profile, early planning and preparation is vital.

Freshman year is generally the time when students are completing general studies courses with a light credit load. This offers the opportunity to add one to two introduction level courses in computer science. After completing a few entry-level courses in computer science, if you
continue to have interest in computer science curricula, you are now positioned to take courses that begin to broaden your I-shape into a T-shaped student over the next few years.

If you are interested in both, accounting and computer science, this creates a favorable situation. You are now positioned to challenge yourself to take courses that broaden the I-shape into a T, while unlocking many career opportunities. Senior year is the conclusion of the curriculum and commonly the time when you are completing your advanced level accounting courses. If you supplemented computer science courses throughout your college career you are now finalizing your strong growth into a T-shaped student. A student that is an accounting major often takes a fifth year to accumulate 150 credit hours to be able to sit for your CPA exam. Taking the enterprise courses and systems testing courses to fulfill your additional credit hours will not only provide you with career alternatives similar to the finance major previously discussed but given the role and cost of IT services in enterprise class businesses it will make you significantly more competitive graduate as compared to a typical I-shaped accounting graduate.

For students in any discipline or major perhaps Jones said it best, "even though a single testing course can never make one an expert, the course can inculcate principles, attitudes and fundamental skills that equip the student for life-long learning" [JONES].

No matter what your major, it is imperative that you maintain a strong GPA. Enterprise tester positions are highly competitive.

**Newly-Hired Employee Activities and Skills**

Across industry, employees are generally considered newly hired if they have joined a test organization for the first time within the last 5 years. Many new hire employees feel pressured to excel immediately upon entering a new environment. For new hires in enterprise test, there is a lot to cover and the process and scope is often overwhelming. While it is not possible to become an enterprise test expert in weeks, months, or in most cases even a few years, following the suggestions herein will help a new hire to stand out from their peers, accelerate their learning curve, and gain both broad and deep skills across enterprise test environments.

Perhaps the most important tip for new hires is to learn your environment. You can not effectively test something that you do not understand. To gain this type of knowledge, ask to help construct or configure your next test environment. Setting up a test environment from the ground up can really accelerate the understanding of the system under test and the various component interactions involved. Plug in the cables, install and configure necessary software, setup your hosts, and disk storage systems. If scripts or automation are to be used during testing, read through the scripts to gain an understanding of what they achieve. If you are not responsible for one or more of these areas, ask if it is possible to shadow or assist the person that is.

As a new hire you will be expected to have questions. However, when you go to others for help, do not just let them fix your problems, instead, ask the relevant questions about the fix itself, and the cause of the issue. Questions such as "Why did you do that?" or "What led you to believe that was the problem" will not only avoid the need for you to ask similar questions in the future but will also enhance your own debug skills. Similarly, when you open a defect, follow it through to closure. By working with development to better understand the debug process, the eventual root cause analysis, and the solution presented, you will deepen your knowledge of both the products and general problem areas to test in future test cycles. These activities build valuable relationships with key development personnel. Further, you will improve your debug skills.
Debug skills are arguably the most important skill a tester can possess. Any tester who is able to debug a problem to the lowest possible level will not only gain respect from peer test engineers and developers, but will also write defects that accurately capture the problem, define the potential scope, and ultimately have the highest probability of getting fixed.

One pit-fall to avoid as a new hire is assuming that it is someone's responsibility to teach you everything. You can't expect to be spoon fed the information you need. Instead, learn to actively seek out the information you need. Resources such as manuals, setup guides, troubleshooting guides, and best practices documents are ideal places to start. By reading these materials you will not only learn more about the products you are testing and their intended uses, but you will also learn about emerging features which may require additional test cases to provide coverage. A tester who is willing to do the extra work and learn the product details will excel and stand out from those who attempt to black-box test every product with the same approach. The better you know and understand the products you are testing, the more complex (and therefore valuable) defects you can find.

To continue on the topic of defects for given products or environments, at the end of every test cycle it is a great exercise to review the list of defects opened against your product during that cycle. When reviewing the list, pay close attention to defect themes and to defects that were opened by other testers that perhaps you overlooked. Learn from this experience and increase your testing coverage and approach during subsequent test cycles.

**Mid-Career Activities and Skills**

Mid-Career test engineers can be at many stages of their career and skill levels can differ depending on that stage. At the mid-career level, a test engineer has probably had experiences working on several products, in several departments and has likely advanced beyond the role of simply executing tests and following a test plan.

Mid-career test engineers are often placed in more of a leadership role where they have a larger responsibility to the successful delivery of the product. This can include being a team lead, a test architect or working directly with development teams to help impact the design of the product. The mid-level career test engineer has a larger role in architecting the test solution and incorporating extensive knowledge of the product as well as customer requirements. Designing an environment that will ensure a quality test is executed is essential to ensure a quality product is delivered.

In some cases, a mid-level engineer may have branched out to work with other areas in the company as part of their regular work. If they have not, this would be a good time to expand their scope and role in the business by going outside their comfort zone to contribute at a higher level in the product development cycle. This would involve collaborating with different organizations to deliver a successful product. For example, a software test engineer might work with a hardware test team to better understand the product and to define potential holes in coverage or ways to optimize for a better return on investment.

No matter where a test engineer is in their career, it is important to continue learning and growing their technical and problem solving skills. The ability to demonstrate high degrees of analytical ability and creativity is essential. These skills allow one to understand and interpret product objectives and market requirements, which in turn help deliver solutions that are competitive in the market.

Keeping the customer at the forefront of test design and execution is essential. Understanding customer requirements and issues becomes imperative when validating product architecture and design. The mid-career test engineers must be able to anticipate potential
problems as new function is being introduced and existing function is being modified and must be able to follow the trends in order to design testing to meet the demands. Many mid-career test engineers have established customer relationships or have contributed to working on different customer issues to help solve customer problems.

It is apparent that customer knowledge is important, but it’s more than just keeping up with requirements and future trends. Following and understanding customer problem trends is critical in being able to define better test strategies and solutions. Improved tests will allow for problem trends to be reversed and in turn improve quality.

In addition to the continued growth of technical skills, soft-skills such as teamwork and communication skills become increasingly important for the mid-career engineer. Being able to share product and technical knowledge, participate and lead in appropriate technical communities, work across teams and organizations, practice and teach good debugging techniques, and design and implement new procedures, guidelines and success measures are all critical. It is apparent to see that as one grows in their career it is more than just technical skills that become critical to the business, the product, the team and the customer.

**Late Career Activities and Skills**

A test engineer in the latter part of their career usually has had much experience and has already noted that technology trends and directions are cyclical. A T-Shaped test engineer should look to new trends and technologies and move their area of depth to a new area that is emerging and given their previous depth areas as well as their hind sight to previous cyclical events they should be able to engage at a high rate of productivity. It is critical that the late career test engineers also work with early career test engineers on projects that will shorten the learning curve of the early engineers. The complexity of today’s enterprise systems is significantly greater than it was twenty years ago and the industry does not have the luxury of having the same amount of time to make progress on the learning curve. One almost has to think of this as not what a late career test engineer can do, but what they can teach to others to scale their knowledge faster across an organization. All too often a late career engineer will just continue on their original depth area and ride out the remainder of their career still making significant contributions in their area but working with their peer group versus working with the early career test engineers. Late career test engineers can also enhance the technical networks of the early test engineers by creating collaborative environments for problem solving with other areas of an organization or parts of the industry.

Late career test engineers need to search out the multiplier leverage points in their knowledge base and work to advance organizations while expanding their knowledge base at the same time. This will provide high value to an organization and enhanced job security.

4. Additional enhancements to Test Careers

**Invention as a career accelerator for enterprise testers**

IBM actively encourages high quality patent submission from its employees, and this opportunity extends to software testers in the same capacity as developers or other members of the IBM business. Virtually all major software corporations engage in patent activities, and these activities extend to those performing software testing practices [MSFT] [ORCL] [HP] [GOOG]. Though invention is not a requisite for promotion or advancement in itself, it is clear that those who perform these activities are those who are advancing the art and science of software
testing, and thus are likely candidates for career advancement within the software test discipline. The incredible emphasis on the protection of intellectual property rights or patents exists as a means to provide competitive advantage in the market place. In addition, the licensing of intellectual property can be a major source of revenue for these software firms. Since the science of software testing is not steeped in decades of invention, there is ample freedom to invent and patent. In general, software companies provide some financial incentive for those who successfully file patents related to their corporate activity, as is the case with IBM. Another reason software titans patent test mechanisms, or other software test inventions, is because it demonstrates a desire to seriously protect ideas which improve the testing process. In many ways this legitimizes, and commercializes software testing as a discipline, and as such countless inventors are working on intellectual property at any time. Note that in many cases IBM opts to donate patents to the common good for the advancement of testing by issuing open disclosures of inventions without submitting them to government patent offices.

Regardless of whether or not the inventions are prosecuted or donated to the software testing community, it is clear, that invention is an aid to career advancement. Those considering careers in software test should be exposed to the fundamental concepts in this space as part of their collegiate educational curricula. Lastly, it should be noted that quite often test engineers patent inventions not directly related to software testing. The process of invention, protecting intellectual property, and career advancement bolstering are the same regardless of the specific area of invention.

**Publishing as an enterprise test engineer**

In addition to patents, many software testers actively engage in various authorship activities as a form of professional or academic service to the software engineering and test communities as well as consumers. Many software testers become so proficient with software packages from their copious interactions with the software under test, that they become recognized as experts. Which in turn results in the authoring of books or papers on the software under test whether that software is middleware, an end user application, or an operating system [LIB][DIRECTOR][ZVM].

Typically this activity coincides with testing, or shortly after testing completes and the software is generally available. Other testers will write about the act of testing itself, by publishing academic papers on the tools they use for testing [AGEDIS], or about the testing process itself [EXP], industry conference papers [TRANS][CHIPS], or authoring entire books on the testing discipline [STT][PATTON][STFPEK]. In addition testers often write about subjects related to the software they test [XEN] [ZLINUX].

**How we intersect with the customers of our products**

Because of the diverse skills and unique perspective on the whole system, system testers are in a unique position to connect with customers and have a direct impact on the success of the customer’s enterprise computing production environment. Some of this is through formal customer engagements, while other ways involve informal analysis of field escapes and problems.

It is important for system test to adopt an “outside in” approach to understanding the customers and follow a closed-loop process in which all information gained from intersecting
with the customers is in turn, incorporated into the test environment. This results in continual test improvement and continual product improvement.

There are many programs that allow test engineers to interact with customers. This can be through informally shadowing a support team to offer their expertise and support in solving problems and designing solutions for the customers. More formal programs allow a test engineer to become an advocate for a customer providing a more personal relationship. Many products include Beta releases which allow a customer to get hands on experience with the product earlier and also interact and work with the test and/or development team during the process. These programs allow test engineers to learn more about the customers concerns, road maps, how they use our products.

The common theme when intersecting with customers is to consider what a client would do with the product, what innovation are they looking for, how it improves their business and what can be done to make it better. Information gained from customer engagements are used to define test scenarios, and design and build customer like test environments. By better understanding how customer use our products, we can perform similar tasks and do things as a customer would when setting up the test floor, defining test scenarios, performing system maintenance, and other system administration activities.

All of these opportunities allow the system tester to stay connected with the customers, further enhance their skills and provide a valuable service to the customers and the company which in turn results in a more valuable T-shaped engineer.

5. Conclusion

The authors of this paper hope that this glimpse into the career path of enterprise testing has proved insightful. It is our belief that the nascent emergence of test career paths in the software and hardware testing disciplines is a logical progression for any enterprise company. While not all companies may elect to implement a model such as IBM's, we invite open and public discussion of the enterprise system test career path such that best practices can be shared in the industry. We simultaneously suggest universities adopt changes to curricula to better prepare students to maximize their career potential in test.
6. References


[GOOG] : http://www.google.com/patents/about?id=Dc3OAAAAEBAJ&dq=%22Assignee:+Google+Inc.%22+computer+testing

[EXP] Experience remote usability testing. Velda Bartek (bartekva@us.ibm.com), Senior Software Engineer, IBM, Deane Cheatham (dcheath@us.ibm.com), Human Factors Engineer, IBM. Availible online: http://www.ibm.com/developerworks/web/library/wa-rmusts1/

[JNPR] Systems Test Distinguished Engineer Job Description 913360. [Available online]: https://careers.juniper.net/careers/careers.html Due to the transient nature of job descriptions the authors can supply a copy of the job description on request from browne@us.ibm.com


[WHIT], Whitaker, J.A. What is software testing? And Why is it so hard? IEEE software 17, 1 (Jan 2000), 70-79.


7. Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:
AIX 5L™
AIX®
z/OS
z/VM
IBM®

The following terms are trademarks of other companies:
AT&T™ AT&T, the AT&T logo and all other AT&T marks contained herein are trademarks of AT&T Intellectual Property and/or AT&T affiliated companies.
Internet Explorer, Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
HP-UX is a registered trademark of Hewlett-Packard Company. (HP-UX®)
Juniper Networks, the Juniper Networks logo, JUNOS, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries.
Linux is a trademark of Linux Torvalds in the United States, other countries, or both.
Oracle and Java are registered trademarks of Oracle and/or its affiliates.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Other company, product, or service names may be trademarks or service marks of others.
8. AUTHORS

Tara Astigarraga
Advisory Software Engineer, IBM Corporation
asti@us.ibm.com
Tara Astigarraga is an Advisory Engineer at IBM in the Systems & Technology Group organization. She works in SAN test for Systems & Technology with focus in enterprise and midrange systems environments and protocol test. Her test responsibilities span IBM’s System x and p servers/blades, IBM storage products (entry, midrange and enterprise storage) and Cisco/Brocade SAN switches. Tara specializes in complex error inject and protocol test for Fibre Channel and Fibre Channel over Ethernet (FCoE) environments. She has been with IBM for 10 years and has held a variety of roles in test and support. She is also an avid inventor, especially in the areas of pro-active error determination and recovery in open systems environments.

Michael Browne
Distinguished Engineer, IBM Corporation
browne@us.ibm.com
Michael Browne (Mikey) is a Distinguished Engineer and Master Inventor at IBM in the Systems & Technology Group organization. He is the Chief Test Engineer for Systems & Technology Group and a member of the IBM Academy of Technology. He is responsible for the system testing of all of IBM’s hardware systems (System z, i, p and x), all of IBM’s designed and OEM offered storage subsystems and the majority of the operating systems that run on those platforms. His expertise is in complex configuration and high availability environments. Mikey spends a significant amount of time with IBM’s clients implementing and debugging IBM products up and down the stack in complex environments. His focus has been mainly in open systems and distributed enterprise services and management of those environments. Due to his 30 years of client facing and engineering test roles he has vast experience in what can go wrong and what to do to resolve it. Mikey has been a career test engineer but has also held positions in IGS system support, IGS and product development management, system design, system architecture and product development.

Eli M. Dow
Staff Software Engineer IBM Corporation, Ph.D. Student Clarkson University,
emdow@us.ibm.com
Eli M. Dow is an author, inventor, and computer scientist working in the IBM Systems & Technology Group. He has written extensively about various virtualization platforms and high-availability reference architectures. His most recent job role was enterprise testing of operating systems and middleware running on IBM system Z mainframe hardware at the IBM Test and Integration Center for Linux in Poughkeepsie NY. Eli is currently on a reverse sabbatical from IBM finishing his Ph.D. research at Clarkson University. His research interests include operating systems, human-computer interaction, and software testing.
Christina Lara
Senior Software Engineer, IBM Corporation
chrslara@us.ibm.com
Christina Lara is a Sr. Software Engineer at IBM in the Systems & Technology Group organization. She is currently the Test Execution Lead for the Scale Out Network Attach Storage (SONAS) product. Overall responsibilities include leadership for test execution for a given release across the various test areas and ensuring test areas are in sync. In addition, identifying problem areas that could have impact on test progress and achieving program milestones. Christina has spent her 10 year IBM career as a test engineer within the STG organization working across several storage products; ESS800, DS8000, XIV and SONAS. In 2008, Christina was able to participate in a one year assignment working closely with IBM Systems Group Chief Test Engineer on customer test related issues, Academy of Technology conference(s), and enhancing her overall IBM product knowledge and testing across several business units. Christina is also an avid inventor in various areas of software and hardware design.

Michael S. Phay
Student University of Maryland Eastern Shore
msphay@umes.edu
Micheal Phay is currently an undergraduate student enrolled in the University of Maryland Eastern Shore. He is currently majoring in Computer Science while minoring in Mathematics and plans to graduate in May 2013. He recently completed a coop assignment as a Test Engineer at IBM STG Poughkeepsie (July 2010 to January 2011) working on system z and the creation of the Introduction to Enterprise System Testing class currently being taught at the University of Maryland Eastern Shore.

Dorian Thomas
Student University of Maryland Eastern Shore
drthomas@umes.edu
Dorian Thomas is a fifth-year senior at the University Of Maryland Eastern Shore, pursuing a Bachelor of Science degree in accounting. Along with the requirements for a B.S. in Accounting, Dorian included IBM infused courses geared towards enterprise system computing and testing to his curriculum. Dorian has recently accepted an offer to join IBM’s Technical Sales Specialist staff and will be located in Raleigh, North Carolina.

Maria R. Ward
Senior Software Engineer, IBM Corporation
mrward@us.ibm.com
Maria Ward is a Senior Software Engineer in the IBM Systems and Technology Group. Maria joined IBM in 1999 and has held several technical test and leadership positions throughout her career, primarily in system level testing for IBM’s AIX UNIX operating system on System p hardware. Maria is currently the Test Architect for the AIX and System p Test Organization with a current focus on IBM’s PowerVM features. She has several direct IBM client relationships which she uses to enhance the system testing of IBM products. Maria is also an avid inventor in various areas of software and hardware design.
Nicholas Wiggins
Student University of Maryland Eastern Shore
ndwiggins@umes.edu
Nicholas Wiggins is a senior student-athlete at The University of Maryland Eastern Shore. He will graduate in May 2011 with a Bachelors of Science in Business Administration with an emphasis in Finance. He has actively been involved in the IBM Initiative at The University of Maryland Eastern Shore. He is also currently enrolled in the Enterprise Testing Course designed by IBM. He has recently accepted an offer from IBM as a Technical Sales Specialist. He will be located in Bethesda, MD.