July/August
2009
Volume 3

SOFTWARE TESTING

This issue of Software Testing delves into mobile application and regression testing.

TechTarget®
The Technology Media
ROI Experts

EDITOR'S LETTER

• Mobile applications and regression testing

→ **WANT JOB SECURITY?** This issue of Software Testing explores the rewards of being thoroughly-modern, thoroughly-trained and always-diligent mobile application and regression testers.

In the mobile application development market, the role of regression testing is proving to be mission critical in the Web 2.0, Rich Internet Application age. Both mobile apps and regression go better with agile and Software Development Lifecycle (SDLC) approaches—as well as some creative thinking—so just being a code wizard or pretty good tester won't get you the gold.

Those who get and stay in the mobile application training circuit could have job security for many years, according to Enterprise Management Associates (EMA) Julie Craig, who wrote this issue's lead article. Just don't expect to quit school at any time, because the mobile technologies will continue to be complex and varied, with new devices appearing faster than bubbles in seltzer water.

Beyond discussing the great career potential in mobile application testing, Craig lays out the mobile app peculiarities that require great development creativity and more savvy testing.

In this ezine's article on testing and retesting defects, David W. Johnson connects the dots between mobile application and regression testing by citing the Blackberry blackouts of 2007 and 2008, which were "an unfortunate example of production issues that should have been detected by a regression test suite in combination with other formal testing efforts."

Helping testers avoid common mistakes is Johnson's key goal in this article. Too often, a user reports a bug; regression testing is done; a "fixed" report goes to the user; and the user experiences the same problem. Johnson suggests practices that can help end this embarrassing scenario.

Regression testers, like mobile application testers, must invest in continuing education due to the constant emergence of new breeds of software, such as mobile applications. This issue is a good first step on that training trail. ■

JAN STAFFORD, Executive Editor
jstafford@techtarget.com

CHAPTER 1

The challenges of testing wireless mobile applications

Mobile application testing requires not just skill, but creativity and resourcefulness. It also requires products and services specifically designed for the challenges of mobile technology. **BY JULIE CRAIG**

→ **MOBILE DEVICES ARE** hot, hot, hot, and applications are the reason why. The lure of always-available Web browsing, email, document viewing and instant messaging has proven to be irresistible to business and personal users alike. Not to be outdone, entertainment-related applications are flourishing as well, with mobile games, Twitter and Facebook, and photo uploads becoming fundamental elements of the new mobile lifestyle. This is reflected in the economy. Annual global sales of mobile phones in units are now roughly equal to sales of laptops, with smartphones accounting for more than 10% of the total.

Vendors are capitalizing on this growth with a host of mobile-enabled applications. Numara® Software, Inc. recently introduced Numara Foot-Prints Mobile, which allows support personnel to access Numara's Service Desk software using the Web browser on their Microsoft® Windows Mobile, RIM® Blackberry®, or Apple® iPhone® devices. TD Ameritrade offers mobile stock trading software which enables a customer to execute stock trades from his or her phone. Workday.com, a Software as a Service (SaaS)-based Enterprise Resource Planning (ERP) provider, recently announced in-development software that will provide access to Workday's Human Resources (HR) and financial applications from an iPhone. Clearly there is strong demand for a growing host of mobile applications, and as a result, mobile development and testing are flourishing.

This explosion can be a bonanza for "mobile literate" development and testing professionals. The outlook for growth in this space is so favorable that job security is virtually guaranteed for the foreseeable future. The bad news, however, is that these jobs are harder than they used to be, thanks to a wide array of handheld hardware, mobile-specific software, and the sheer force of constant change. Mobile applications isn't your grandmother's COBOL, and testing them requires not just skill, but creativity and resourcefulness. It also requires products and services specifically designed for the challenges of mobile technology.

MOBILE APPLICATIONS: SIMILAR TO CONNECTED APPLICATIONS

Even without mobile technology, software development is a notoriously failure-prone activity. I don't need to go into the details since everybody reading this is likely aware of the high percentage of software development projects that fail to deliver in terms of functionality, budget or schedule. Mobile applications add to application complexity, and in doing so, increase development and Quality Assurance (QA) risks.

Mobile development requires the same Software Development Life Cycle (SDLC) approach that adds discipline and governance to traditional software development. Agile methodologies are rapidly replacing older Waterfall processes to become the "de facto" standard governing the SDLC. Agile principles include incremental and iterative cycles, testing early and often, and incorporating end-user input throughout the lifecycle, and they appear to be paying off. BMC Software adopted Agile starting in 2004, and were able to increase product time to market by two to three times over the industry norm (go to www.enterprisemanagement.com[1] for more information on this story).

From the perspective of mobile applications, Agile methodologies are often also the best choice, primarily because of the ongoing focus on customer involvement and iterative testing. The steps required—requirements gathering, design, development, testing, and deployment—are similar for both connected and wireless applications. However, the lifecycle must be tweaked for mobile applications, and it is important to answer some key questions from the outset.

They include:

- Which mobile devices and operating system versions will this application support?
- How do we test applications to make sure they run on those platforms?
- What modifications must be made to accommodate the differences among platforms?
- How will industry innovations be supported going forward, since new mobile devices, technologies, and applications are constantly being introduced?
- How will development and testing processes accommodate the inherent differences among wireless network protocols and mobile service providers?
- How do we know how much testing is enough?

[1] "Agile Management—What Managers can Learn from Agile Methodologies"

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

Each organization will have to answer these questions internally, based on their overall goals and objectives.

KEY DIFFERENCES

Challenges with mobile testing tend to center around several key areas. One is the wide variety of mobile platforms and the complexity around testing an application's compatibility with all supported platforms. Another is the challenge of developing applications that compensate for the facts that the mobile device is not always on and that the wireless connection may not be as reliable as a wired one. Another key consideration is the uniqueness of each user. Customers subscribe to different carrier networks and have different use cases. The "last mile" to their device is likely to run at different speeds and over slightly different protocols depending on user location, roaming, vendor, network, and device. The bottom line is that, if you and I are running the same mobile application, our experiences might be very different. Factoring such variations into expected testing outcomes can become quite complex.

Mobile development requires the same Software Development Lifecycle (SDLC) approach that adds discipline and governance to traditional software development.

The primary problem is one of testing every possible scenario, given that few development organizations own one of every device, version, and carrier connection. And even the most talented QA team will probably not be able to simulate every combination of these variables, especially when the idiosyncrasies of users are factored in. Fortunately, a lively marketplace has arisen around mitigating these challenges, and some of the products and services detailed below may be helpful to organizations struggling with mobile testing.

HETEROGENEITY CHALLENGES AND VENDORS' RESPONSES

Say what you will about Microsoft, for those of us who remember the "old days" when operating systems (OSes) were virtually vendor-specific, the ubiquity of Windows has certainly made it easier to develop and test. Of course, Apple is, and always has been, an exception to the standardization rule, and many Independent Software Vendors (ISVs) develop applications for both Windows and Mac OS. This standardization has been a boon in terms of limiting platform proliferation. Assuming support for both vendors and a given number of operating system levels for each, the worst case testing scenario is still fairly limited.

In contrast, mobile computing is just about where personal computing

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

was 20 years ago. Let's see, you have Symbian, Blackberry OS, Windows Mobile, Palm OS, Apple OS X, Linux, etc., etc. Add to this the fact that there are multiple browsers and little standardization across devices, throw in 2 or 3 release levels each and the result is a fairly complex testing scenario.

Mobile applications isn't your grandmother's COBOL, and testing them requires not just skill, but creativity and resourcefulness.

This can be a problem for testing teams since, unless they have one of each combination of device, OS, version, etc., it is virtually impossible to ensure that a given application works "everywhere". Multiple vendors are addressing this challenge with a wide array of solutions.

Some solutions are aimed at in-house testing:

- **Hewlett Packard** (HP): HP is integrating mobile application testing into its existing Quick Test Professional (QTP) solution by partnering with vendors that specialize in mobile testing. The first partnership is with Jamo Solutions, which offers an end-user experience management solution for Windows Mobile devices. HP is working on other partnerships as well, including one with DeviceAnywhere. DeviceAnywhere markets both cloud-based and on-site versions of its mobile Quality of Service (QoS) solution with support for more than 2000 handsets and 30 mobile operators.

- **Microsoft:** Microsoft is a rich source of testing resources for Windows mobile applications, with a broad range of tools supporting development and testing of wireless applications built using Microsoft technology. Microsoft's mobile development site is a good starting point.

  Resources include a set of mobile emulator images that can be used with or without Microsoft Visual Studio and provide a testing platform for the Windows Mobile operating system.

- **Dexterra™**: Dexterra is a different product entirely, and its flagship product is the Dexterra Concert mobile application development platform. Dexterra is available in two versions. "Enterprise Edition" is an open, standards-based development platform designed to be deployed within the enterprise. "Carrier Edition" is targeted at service providers who plan to host the solution as a service. Both platforms support a wide range of mobile devices as well as software requirements management. Dexterra promises to streamline and simplify development of mobile applications, and may be well worth evaluating for companies which are heavily invested in mobile development.

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

There are also a variety of options for Quality Assurance (QA) managers looking for hands-on testers for specific devices or to supplement in-house teams. This can be a lifesaver when testing schedules back up, or to certify solutions against actual devices instead of emulations. Two vendors offering such services include:

- **uTest®:** uTest is positioned squarely in the software testing realm, and recently announced support for mobile applications. The company's business model is a distinctive one. It consists of a host of software testers (more than 14,000 worldwide) who test mobile applications across a variety of operating systems and platforms. Testers get paid "by the bug". They execute testing scenarios and report unexpected issues to the customer, who then decides whether the issue is, in fact, a defect. If so, the customer pays the tester who found the bug. If not, the customer does not pay.

This model can be valuable because human testers actually simulate the kinds of issues that are likely to arise once the software is released to the public. It is also cost effective, in that uTest's customers don't pay for testing "features" the tester interprets as "bugs".

- **Keynote™ Systems, Inc.**: Keynote is one of the best known Web testing companies in the world, due in part to its frequent press releases detailing performance of well-known online vendors. Click here for an example of a recent Keynote release.

Keynote has now expanded into mobile testing as well, leveraging testers from locations worldwide to deliver an impressive breadth of solutions. They include:

- **Mobile Device Perspective (MDP):** Provides feedback on how effectively mobile devices interact with their service content
- **Mobile Application Perspective (MAP):** Provides information about how well mobile sites function during transmission and delivery
- **SITE Test System:** Provides mobile operators with metrics that quantify the health of the mobile network
- **Global Roamer:** Provides an in-depth understanding of how applications function from various points across the world, based on testing from multiple locales
- **Web Effective for iPhone:** Platform to run large-scale, task-based iPhone Web usability studies

PROBLEMS AND PAY-OFFS

There are obviously many more vendors and solutions in the mobile market than there is space to discuss them. The point to think about is the idea of using automation wherever possible to add efficiency to mobile development and testing.

In recent years, the cost of IT has risen to the point where, for some companies, it is limiting the ability to grow and compete. Software development, for many companies, is a cost

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

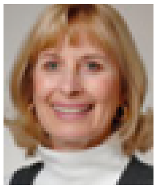
center within IT, and efficiencies in the development and testing spaces are as important to overall cost reduction as efficiencies in the data center.

That being said, rising IT costs aren't due to IT specialists, developers, or testers who waste time. They are due to technology complexity. The past 10 years have seen an explosion of heterogeneous technology, with each new technology requiring specialized personnel, tools, and infrastructure. Mobile is no exception.

Mobile applications represent a "new reality" in terms of applications, in that many of the applications currently under development cannot be 100% tested by in-house personnel and resources. They will be used by literally millions of users, and it is impossible for a testing team with finite resources to duplicate every use case, platform, network, and device.

Because of this, investments in automated testing products and augmentation of in-house test teams with third-party vendors will likely pay off in terms of both testing efficiency and quality. Research has repeatedly shown that software bugs detected early in the lifecycle are much cheaper to remedy than those found in production. Delivering quality software is one way that QA teams can directly impact costs, and this has become even more true with the proliferation of mobile applications. ■

Mobile applications represent a "new reality" in terms of applications, in that many of the applications currently under development cannot be 100% tested by in-house personnel and resources.



ABOUT THE AUTHOR:
JULIE CRAIG, a senior analyst at EMA, has over 20 years of experience in software engineering, IT infrastructure engineering and enterprise management. Julie's areas of focus currently include best practices, configuration management, application management, service-oriented architecture (SOA) and Software as a Service (SaaS). Her goal is to interpret industry trends with a common sense approach and a simplified outlook.

CHAPTER 2

Regression testing hot spots: Coverage, common mistakes

This article discusses regression test opportunities, coverage policies and development, as well as filling you in on five common regression testing mistakes. **BY DAVID W. JOHNSON**

→ **THE REGRESSION TEST** is one of the most misunderstood software testing concepts. Ineffective regression test policies lead to increased post-deployment risk and resulting business losses.

Several recent software failures should have been detected and mitigated by effective regression test suites. For example, the Blackberry blackouts of 2007 and 2008 are an unfortunate example of production issues that should have been detected by a regression test suite in combination with other formal testing efforts. The convergence of adaptive development practices (agile, extreme, etc.) and complex integrated application spaces (Internet, cloud computing, etc.) create a growing risk profile that needs to be mitigated by a disciplined implementation of effective regression testing policies.

Testing early and often significantly reduces remediation costs while mitigating potential production risks. In this article, I will walk you through regression test opportunities, regression test coverage policies and regression test development, as well as filling you in on five common regression testing mistakes.

Several recent software failures should have been detected and mitigated by effective regression test suites.

Regression testing is often referred to as a Test Phase or Test Type, but it is really the execution of one or more tests within a given Test Phase. For the purposes of this article, a regres-

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

sion test is the execution of tests to verify that changes in the application landscape or architectural framework have not negatively impacted existing, unchanged functionality.

Generally, according to WhatIs, a regression test involves the execution of tests in a current Test Phase that validates the changes in the current application landscape—such as software, hardware, data, and/or metadata—have not negatively impacted existing, unchanged functionality.

REGRESSION TEST OPPORTUNITIES

In the past, a regression test has been closely associated with the Function Test Phase and System Test Phase. For the most part, the IT community has come to realize that testing early and often significantly reduces remediation costs while mitigating potential production issues. The following table illustrates regression test opportunities as the solution moves from being conceptual to a production release/package. The solution set can include software, hardware, data, metadata, or a mixture of these and any other aspects of the application landscape.

For the most part, the IT community has come to realize that testing early and often significantly reduces remediation costs while mitigating potential production issues.

In the chart below, *Solution Set*

SOLUTION SET IS...	EXISTS AS...	REGRESSION OPPORTUNITY...
Conceptual (no code)	A Model	**Design Review Phase**
Under Construction	Discrete Units	**Unit Test Phase** • *Instrumentation*
Constructed	Functional Components	**Function Test Phase** • *Functional Test Sets*
Functional	Integrated System	**System Test Phase** • *System Test Sets*
Confirmed	Package or Release	**User Acceptance Test Phase** • *UA Test Sets*
Released	Production Solution	**Production Monitoring**

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

refers to the current maturity of the solution set. *Exists as* is the current deliverables supporting the solution set. *Regression Opportunity* notes the available regression test opportunities. The *primary opportunities* are in italics.

■ Unit Test Phase: *Instrumentation*

The first opportunity to perform substantive regression testing occurs in the unit test phase. The primary actor responsible for regression testing within the Unit Test Phase is the developer. The developer accomplishes this by implementing the Adaptive development principles—namely agile, extreme and rapid application development (RAD)—of continuous integration and code instrumentation. Remediation of defects at this level of solution maturity will yield significant downstream benefits and significantly reduce remediation cost.

I have recently had the pleasure of working with several disciplined agile development teams that have fully implemented code instrumentation and continuous integration. The results of their efforts were a significant decrease in defects later in the development lifecycle.

■ Function Test Phase: F*unctional Test Sets*

The second opportunity to perform substantive Regression Testing occurs in the Function Test Phase – this is the opportunity that is most often leveraged by organizations. The primary actor responsible for Regression Testing within the Function Test Phase is the Tester. The Tester should accomplish this by implementing a suite of automated scripts that test the functional areas of the application deemed to be in-scope for the current test engagement—these automated scripts should be constructed to be reusable test artifacts. Remediation of defects at this level of solution maturity will reduce remediation costs.

There is often not enough time in the test schedule for significant manual regression testing.

Manual regression testing can also be performed, but the overall return on investment is greatly reduced. Manual tests take longer to execute and are prone to failure over multiple executions. There is often not enough time in the test schedule to allow for significant manual regression testing.

■ System Test Phase: *System Test Sets*

The third opportunity to perform substantive regression testing occurs in the System Test Phase. The System Test Phase should include several types of testing; from a Regression testing perspective, the two most important would be Business functionality and performance.

The primary actor responsible for regression testing business functionality within the System Test Phase is the tester. The tester should once again accomplish this by implementing a suite of automated scripts that test the business functionality deemed to be in scope for the current test engagement; these automated

Avoiding Common Regression Test Mistakes

THE MOST COMMON regression test mistakes come from misplaced assumption that regression tests:

- **ARE STATIC**
  - » **Regression test maintenance must be ongoing. Regression tests need to be maintained as IP (intellectual property) that meet the current organizational test coverage needs.**
  - » **I have worked several engagements in the last few years that have treated existing regression test suites as static artifacts. In one case, the client ended up with over five-thousand test cases that could never be executed within the time-frame provided for testing. Once we did a maintenance sweep of these manual regression test scripts we had less than one-thousand automated test cases.**

- **ARE A NICE-TO-HAVE**
  - » **Regression testing is only a nice-to-have if the current organizational coverage needs—such as regulatory, IT governance and risk—does not require any supportive regression test metric. This would be a rather unusual circumstance, but it does occur; i.e., an in-house reporting tool.**
  - » **Anyone who works in the testing space has seen the consequences of this mindset. In several cases I have seen clients do little or no regression testing and pay the cost in post-production support; a cost that far exceeds the cost of actually doing the testing.**

- **NEED TO RETEST EVERYTHING**
  - » **This is usually not a practical goal unless the nature of the application space (i.e. medical devices) and the level of risk, justify a complete end-to-end regression test—the testing team simply runs out of time. Unfortunately this usually means that the “easier” tests are executed first while the “harder” regression tests are deferred. Ensure you create a test schedule that fits into the available Regression test time-box and addresses the higher priority coverage items first.**

↘
EDITOR’S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

scripts should be constructed to be reusable test artifacts. Ensure these tests do not replicate other function testing efforts.

The primary actor responsible for regression testing performance within the System Test Phase is the performance assurance test engineer. The performance assurance test engineer should accomplish this task by implementing a regression suite of performance tests and associated performance monitors—these suites should be constructed to be reusable performance test artifacts.

Manual regression testing of business functionality can also be performed but the overall return on investment is greatly reduced. Manual tests take longer to execute and are prone to failure over multiple executions. There is often not enough time in the test schedule to allow for significant manual regression testing.

User Acceptance Test Phase

The final opportunity to perform regression testing occurs during the User Acceptance Test Phase. These manual tests are performed by the end-user. The primary purpose of regression testing during this test phase should be assuring the end-user that no unexpected changes have occurred in previously existing functionality.

REGRESSION TEST COVERAGE

Regression test coverage policies guide the organization in determining what regression test artifacts are required to be built and, more importantly maintained. Regression test policies should be based on three key factors:

1. Regulatory Space

- Rules, Regulations, Laws, and Standards that apply to the application space.
  - i.e. FDA, FAA, SOX, etc.

2. IT Governance

- Corporate governance as it applies to the regulatory space.
- Corporate governance as it applies to corporate rules, regulations, and standards.

3. Risk Profile

- The acceptable level or risk for any given aspect of the application space.

Regulatory Space

The regulatory space can have a significant impact on the scope of Regression testing within any given test phase. Mangers and test leads responsible for testing need to ensure they understand their legal and fiduciary responsibilities within the regulatory space and the impacts of this responsibility on the overall scope of the testing effort.

IT Governance

IT governance implies a system in which all stakeholders have the necessary input into the IT decision making process preventing IT from independently making and later being held solely responsible for poor decisions.

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

Therefore a board needs to:

- understand the overall architecture of its company's IT applications portfolio;
- ensure that management knows what information resources are out there;
- know what condition information resources are in;
- understand what role information resources play in generating revenue.

There are several definitions of IT governance:

- **Weill and Ross** focus on "Specifying the decision rights and accountability framework to encourage desirable behavior in the use of IT."

- **IT Governance Institute** expands the definition to include foundational mechanisms: "… the leadership and organizational structures and processes that ensure that the organization's IT sustains and extends the organization's strategies and objectives."

- **AS8015**, the Australian Standard for Corporate Governance of Information and Communication Technology (ICT) defines Corporate Governance of ICT as "The system by which the current and future use of ICT is directed and controlled. It involves evaluating and directing the plans for the use of ICT to support the organization and monitoring this use to achieve plans. It includes the strategy and policies for using ICT within an organization."

From a regression testing perspective application of IT Governance to the application landscape means the appropriate level of information must be consistently supplied to support decision making at the board level. Regression testing provides several opportunities to collect appropriate information—as IT Governance practices mature the amount of information collected needs to be assessed for completeness.

■ Risk Profile

Risk is the net negative impact of a vulnerability taking into consideration both the probability and impact/cost if the risk becomes an issue/event. Risk management is the process of identifying risk, assessing risk, and taking steps to reduce risk to an acceptable level. Risk management often encompasses three main processes: risk assessment, risk mitigation, and risk evaluation—Regression testing deals with supplying data

Regression tests should be consistently reviewed and adapted to meet the current coverage needs.

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

to support risk evaluation. The organization should have already considered what level of risk is acceptable for a particular application. It is the responsibility of the organization to understand the basis for the current risk assessment and apply the appropriate level of Regression testing rigor, to meet the specified level of acceptable risk.

Acceptable levels of risk can be categorized as:

TIER 1: Critical Application Space

- Minimal or no interruptions in service are acceptable

TIER 2: Important Application Space

- Short uncommon interruptions in service are acceptable

TIER 3: Normal Application Space

- Short common interruptions in service are acceptable

TIER 4: Non-Critical Application Space

- Interruptions in service are acceptable
- Application can be pulled

Regression Test Policies: Coverage

1. Regression test coverage in combination with other formal testing efforts shall provide sufficient test coverage to satisfy the Regulatory Space.

2. Regression test coverage in combination with other formal testing efforts shall provide sufficient test coverage to satisfy the current IT Governance requirements.

3. Regression test coverage in combination with other formal testing efforts shall provide sufficient test coverage to satisfy the current acceptable Risk Profile.

REGRESSION TEST DEVELOPMENT

Regression tests should address all meaningful:

- **Code Branches:** Regression testing at the level of Unit Test
- **Functional Business events:** Regression testing at the level of Function Test
- **End-to-End Business events:** Regression testing at the level of System Test
- **User Experience:** Regression testing at the level of User Acceptance Test

In each case appropriate tests need to be designed, built, and constructed. These tests need to be easy to maintain, with definitive pass/fail conditions, and when possible automated.

Regarding regression test automation, the User Acceptance test should be defined and executed by representatives of the user community - these types are rarely automated. All other levels of regression testing should be either automated using test automation tools or instrumented using Adaptive development techniques.

On the maintenance front, regression tests are often treated as a static

↘
EDITOR'S LETTER

↘
CHAPTER 1
THE CHALLENGES OF TESTING WIRELESS MOBILE APPLICATIONS

↘
CHAPTER 2
REGRESSION TESTING HOT SPOTS: COVERAGE, COMMON MISTAKES

set of artifacts to be applied during each release cycle. Regression tests should be consistently reviewed and adapted to meet the current coverage needs (regulation, governance, and risk). This review should include adding to, removing, and merging regression tests in the regression test suite. If test case maintenance is not incorporated into Regression testing, the tendency is to end up with an unmanageable number of tests that may not address the coverage need.

In software development, there are some key regression test policies that you can't overlook without bad consequences. From my experience, these are the most important:

- **Regression tests shall be automated using test automation tools or instrumented into the solution using Adaptive development techniques.**

- **Regression tests shall undergo scheduled maintenance designed to right size the Regression test suite to meet the current coverage needs, such as regulation, governance and risk.**

- **Regression test dependencies—plans, schedules, test data, test environments, tools, etc.—must be considered part of the regression test suite.**

Regression testing needs to be incorporated into the overall testing strategy. The goal should be to mitigate risks to production caused by unexpected impacts against unchanged functionality; creating regression test "saves" early in the system development life cycle. A "save" is the detection of a defect before it reaches production thus providing the opportunity to remediate the defect before it is released. The regression testing effort also needs to be right-sized to meet the corporate coverage requirements and fit into the available regression testing time-box. ■

ABOUT THE AUTHOR:
DAVID (DJ) W. JOHNSON is a senior test architect with over 22 years of experience in Information Technology across several industries having played key roles in business needs analysis, software design, software development, testing, training, implementation, organizational assessments, and support of business solutions. Developed specific expertise over the past 12 years on implementing "Test ware" including - test strategies, test planning, test automation (functional and performance), and test management solutions.