

The Crisis of Complexity

A Wind River Market Survey of Device Software Testing Trends and Quality Concerns in the Embedded Industry

Table of Contents

Executive Summary	2
Survey Methodology	4
Survey Topics and Response Summaries	5
Participant Roles and Industry Segment	5
Product Complexity Driving New Testing Challenges	6
Shorter Testing and Defect Resolution Cycles	10
Inadequate Visibility into Quality	13
High Cost of Poor Quality	17
Investing in Test Automation	21
Survey Statistics Summary	23
About Wind River	24

Executive Summary

Device market dynamics, including skyrocketing software content and new architectural complexity, are challenging development and test teams in every industry segment. Quality assurance groups are faced with increasingly short delivery schedules, despite having significantly more work to do. Management does not have the quality visibility they need to make confident release readiness decisions. The penalty for late-cycle quality surprises and shipped defects is extremely high. Companies across all segments are investing in new test tools with an emphasis on more automation, better measurement, and new run-time analytics technologies that can help them cycle faster and improve the thoroughness of testing for their complex embedded software.

Leveraging its extensive market reach, Wind River conducted a survey of the state of software quality testing in the embedded device industry during the months of April and May 2010. Nearly 900 software professionals from all major industry segments responded to provide their feedback on their challenges, trends, and needs.

Product Complexity Driving New Testing Challenges

Survey respondents highlighted two realities that are clearly impacting their day-to-day operations. The first of these is the dramatic increase in the amount of software content (in terms of lines of code) that is now typically included in embedded devices. The software content is doubling about every two years. The sheer volume is making it increasingly difficult for QA and test teams to keep up with traditional tools and processes.

The second major issue is the rapid increase in architectural complexity. This includes moving from 16-bit to 32- and 64-bit architectures, greater utilization of multi-core technologies, and ongoing operating system changes. Other issues include the use of multiple operating systems in a single product, advanced user interfaces and networking, and virtualization technologies.

These newer systems and technologies pave the way for higher performance and greater capacity, but they significantly increase the overall complexity of embedded devices. With greater complexity comes the need for more and better testing.

Shorter Testing and Defect Resolution Cycles

A majority of survey participants reported that market conditions have forced them to shorten their development schedules by as much as 18 months.

Many companies are turning to iterative methods to improve feedback within their product cycles. Some are using agile techniques. However, most respondents are still heavily using manual techniques for system testing and especially for more complex testing of error conditions and performance validation. Defect isolation and repair cycles are typically measured in days per bug. Together these bottlenecks increase the need for new approaches to test automation.

Inadequate Visibility into Quality

When asked how they measure software quality today, survey respondents most often cited metrics that are reactive in nature such as tracking customer-reported failures and open defects rather than metrics that can help them prevent defects. These metrics, especially the customer-reported problems, count how many “horses have already left the barn.”

Executives, QA, and test professionals at embedded companies know that you can't fix what you can't measure. They also know that the answer lies in more thorough testing done more often. Survey participants clearly indicated that they have a strong desire to be more proactive, but their responses also reveal a significant disparity between their intentions and their actual testing.

The survey uncovered a significant disparity between testing goals and reality (as measured by test coverage analysis), where a minority of respondents actually have access to the information they need to assess quality. Respondents further reported that often the software quality information they do receive is later proved to be wrong, leading them to make product readiness decisions based on inaccurate information.

The disparity between testing goals and actual results, the management blind spots regarding software quality levels, and the decision making based on bad information are all contributing to a crisis in confidence in software quality in the embedded industry.

High Cost of Poor Quality

Respondents show that the cost of their testing investment is typically 20%–40% of their program budget. But the true cost of poor quality is much higher. It is measured by damage to company brand and lost revenue due to missed market windows.

Participants are suffering from repeated schedule slips due to late-cycle quality surprises, though a significant number make the decision to ship defective products anyway and fix them later. These defects are not in standard features that they test for but from unanticipated uses of the product that are not tested.

Investing in Test Automation

Given the high cost of product failure, accelerating complexity, and reduced schedules, the industry is turning to test automation. To help address these problems, 68% of respondents plan to automate more of their test processes in 2010. They are moving to new tools that can help test teams and their management better understand how well they are testing, better focus their efforts on the areas needing testing, and reduce cycle time through more automation.

Survey Methodology

The purpose of the survey was to gain a detailed snapshot of how executives, development managers, software quality assurance (QA) and test team leaders, and other involved staff currently view the embedded device software quality test landscape. Recent changes and new challenges and strategies for managing them were of particular interest.

To get that picture, Wind River's device test product division fielded a four-part survey to lists of individuals who work for embedded products companies. In total, nearly 35,000 individuals in North America were invited to participate in the survey via email. Participation was also promoted and encouraged on the Wind River website and on executive blogs.

Survey participants included a representative mix of company executives, software developers and development team managers, and QA and test team leaders. Responses are only reported in aggregate because the survey was conducted anonymously.

Survey Topics and Response Summaries

Participant Roles and Industry Segments

While 47.8% of respondents identify themselves as software developers, another 46.4% have roles in management and test (including executives, directors, managers, project leads, quality officers, etc.).

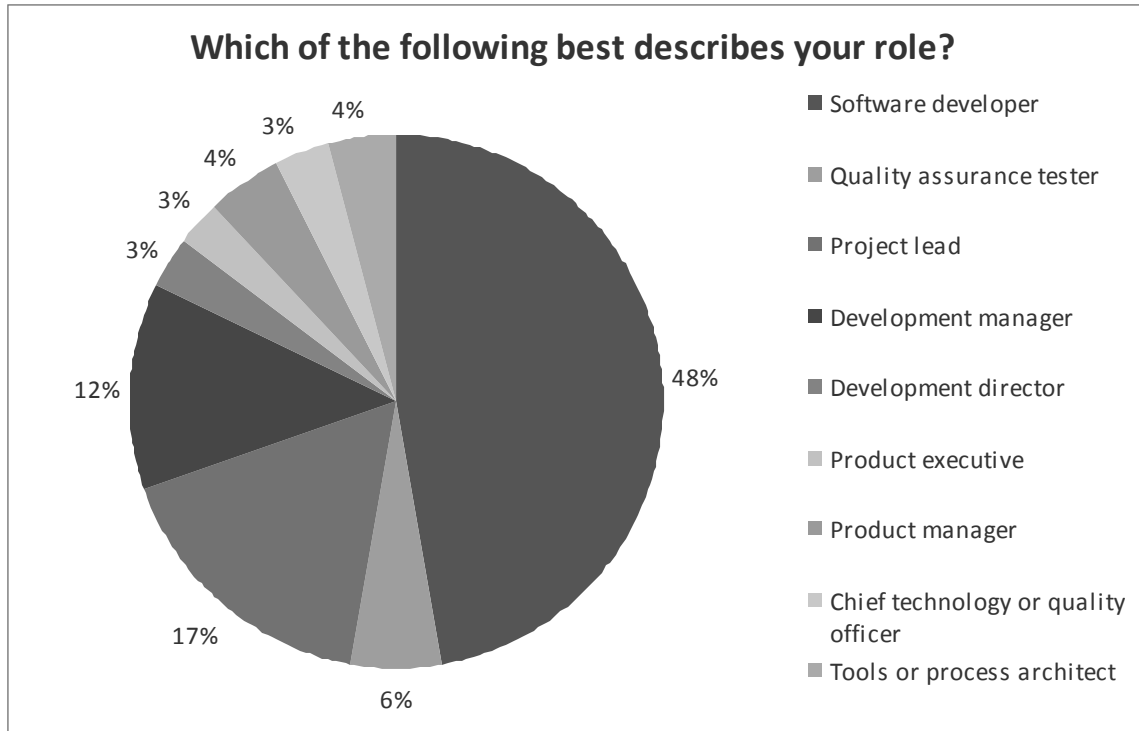


Figure 1: Participant roles split between individuals and management

Survey respondents work across the largest embedded industry segments. The highest participation is in the aerospace and defense, networking equipment, and industrial product verticals.

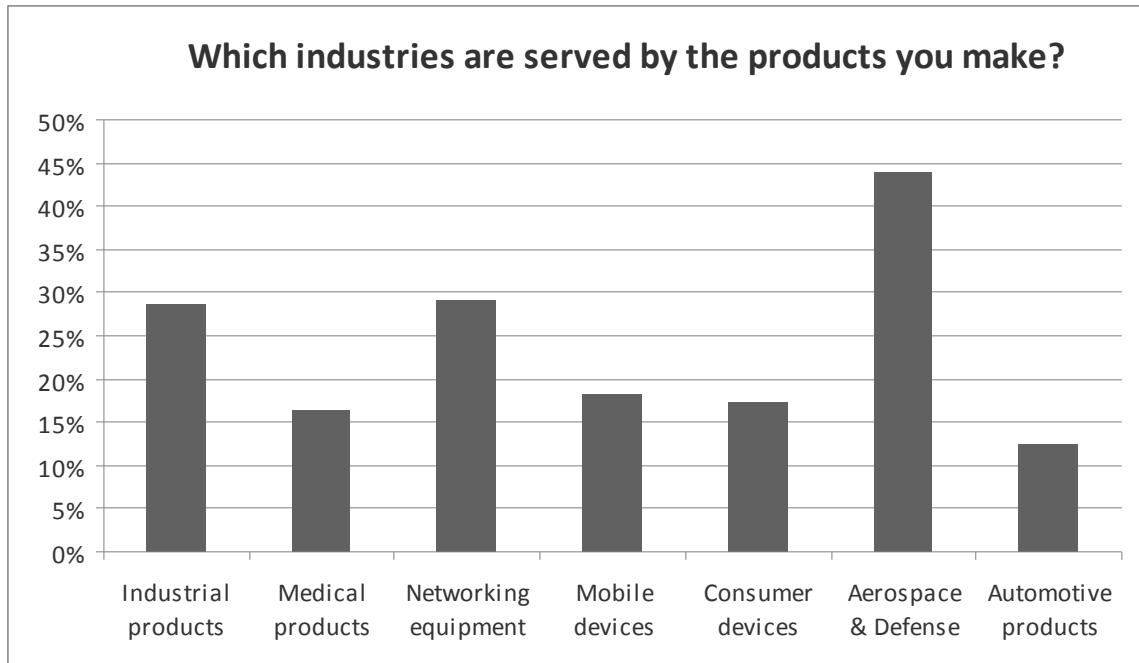


Figure 2: Full range of industry segments represented

The sizes of companies from which responses were received were diverse and reflective of the industry in general. About 50% are companies with 1,500 or fewer employees, and 50% are companies with more than 1,500 people. About 41% of the companies have more than 5,000 employees. A total of 54.1% of participants identified themselves as Wind River customers. More than 40% of respondents said they were not customers.

According to responses, the person who is ultimately responsible for the quality of delivered products was evenly divided among QA manager (29.6%), development manager (25.3%), and vice president of engineering (18.1%). While testing and QA personnel were noted as key players, about 50% of respondents indicated that quality responsibility primarily belongs to development management, vice president of engineering, the chief executive, or “everybody.” In the embedded industry, development plays a large role in testing and integration of the product.

Product Complexity Driving New Testing Challenges

Software not only makes up a huge portion of device features but software content is also experiencing huge growth rates. More than 54% of participants indicated that the software they are incorporating in their devices increases between 25% and 100% annually.

Their devices are no longer “fixed function” but are platforms that require periodic upgrades to the software, according to 96% of respondents. Upgrading is done on an “as needed” basis by 47%; and 48% update on a regular basis, ranging from less than six-month to more than 18-month cycles.

Participants identified several key issues as roadblocks to on-time delivery of software. The top operational factors rated by respondents as the biggest challenges to their ability to deliver software on time include the following:

- 1) Changing requirements
- 2) Becoming familiar with new technology (processors, operating systems, etc.)
- 3) Integrating and debugging devices
- 4) Complex system design
- 5) Time allotted for testing

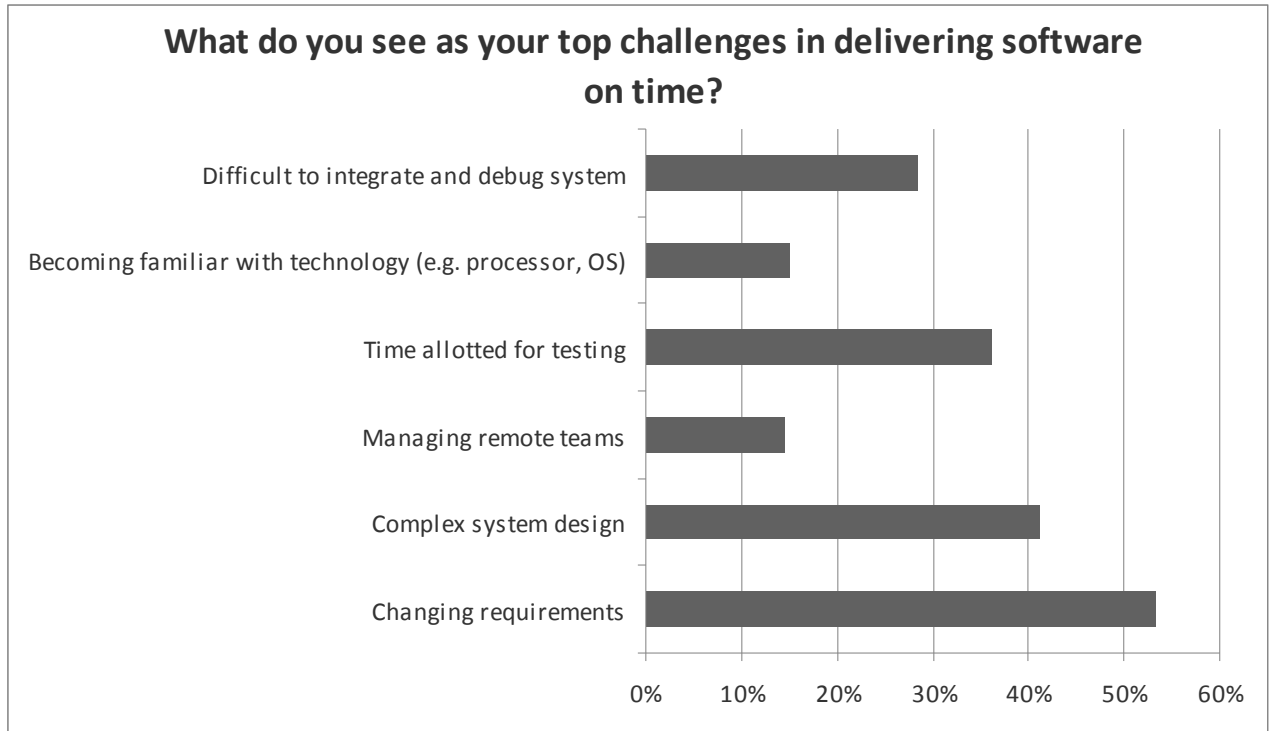


Figure 3: Top challenges to on-time software delivery

The survey also asked about top challenges affecting participant's software testing processes. Participants identified a range of issues, the following being the most frequently cited:

- 1) The amount and complexity of code they are required to test
- 2) Complexity of system architecture
- 3) Insufficient test automation
- 4) Moving to new operating systems

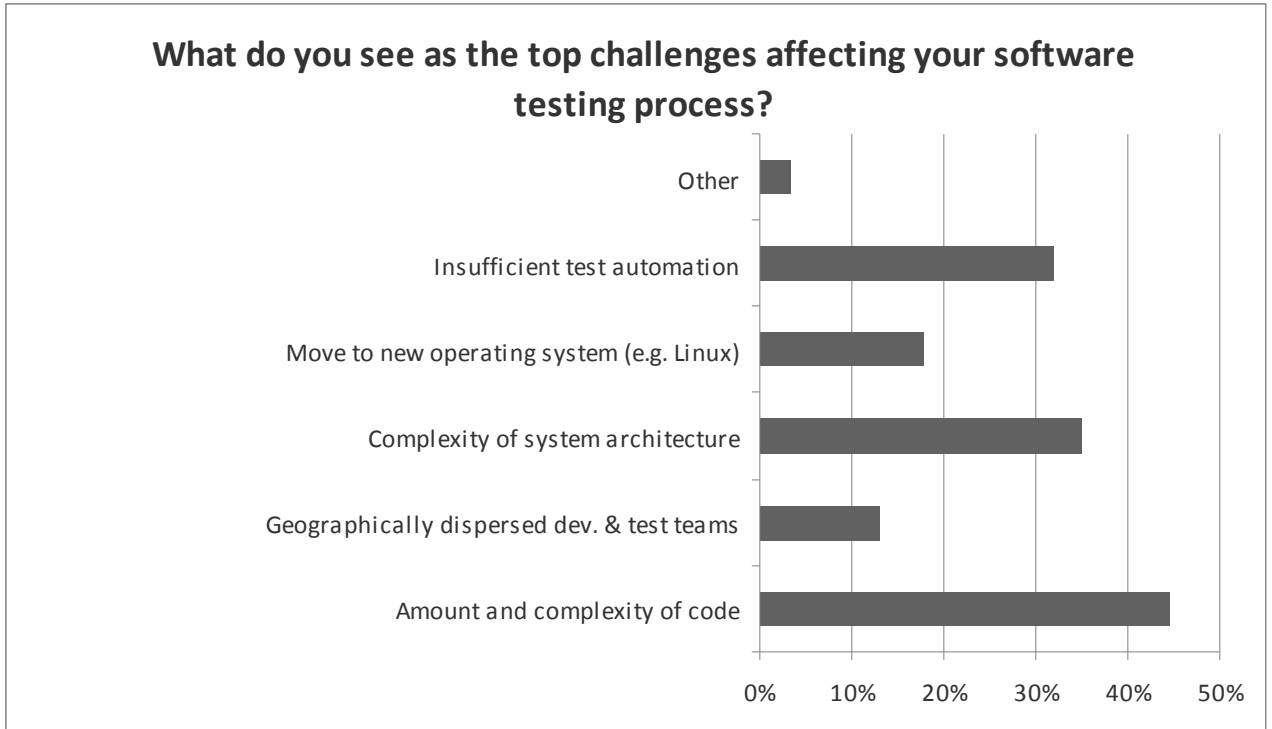


Figure 4: Complexity, change, and lack of automation rated as top challenges in software testing processes

Insight into why these trends are challenging can be gained from responses to a question about which architectural trends are contributing to testing complexity. Not surprisingly, changing operating systems was the top-rated architectural challenge followed closely by transitions to multi-core processors.

Other often cited challenges include advanced user interfaces, advanced networking functions, and the switch from single to multiple processors. Although it was not ranked in the top five, the use of virtualization technologies, cited by 23.9% of respondents, is clearly a growing challenge in the software quality testing world.

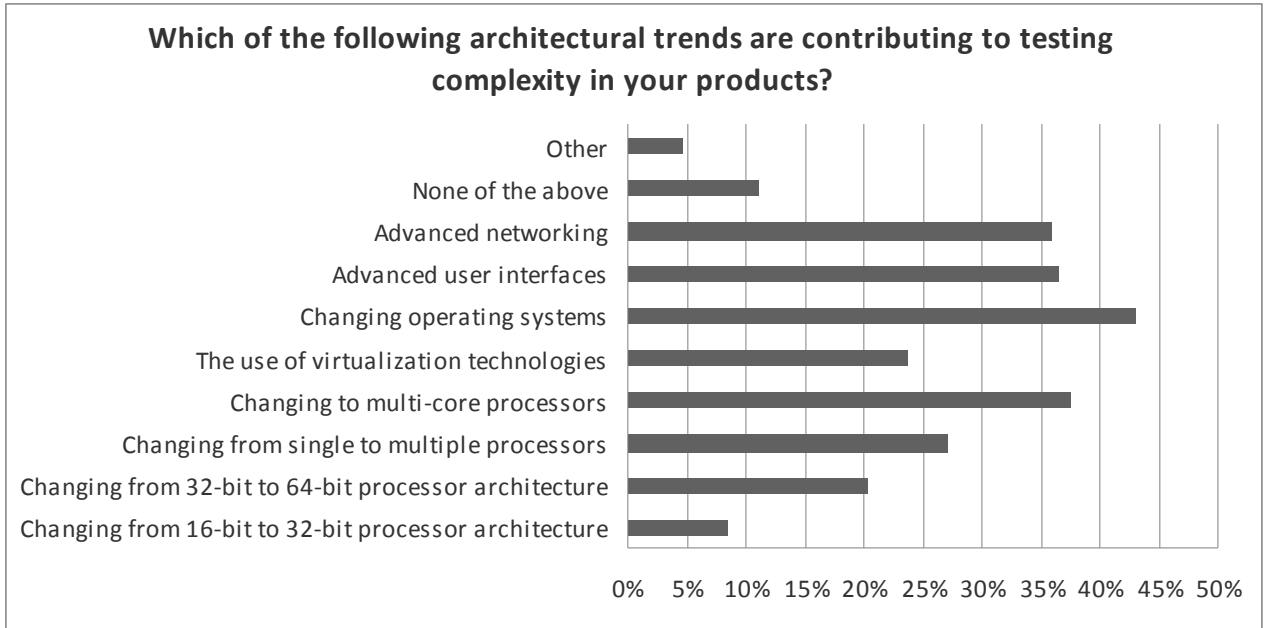


Figure 5: Platform changes and advanced technology are top challenges for testers

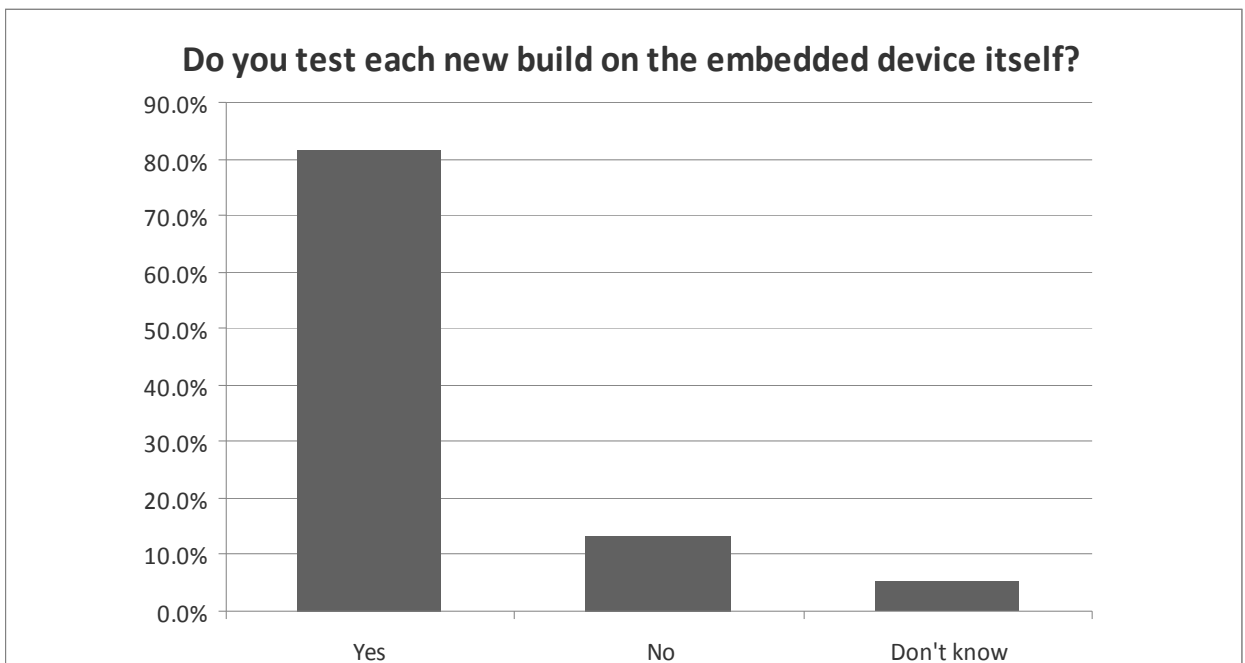


Figure 6: Testing on the embedded device is critical

More than 80% of respondents report that testing on the embedded device itself is critical, and each new build is tested on the device. Many respondents commented on how challenging embedded device testing is.

“Most embedded analysis tools impact application timing, which makes them less than useful. I would be interested in tools and technology that make this less of a problem.”

“Technology (embedded +OS) along with architectures and end applications are getting ever more complex, making test and qualification a challenging task.”

Shorter Testing and Defect Resolution Cycles

A majority (64%) of respondents indicate that market conditions have forced them to shorten their development schedules. In terms of the length of their development cycles, a majority of respondents (57%) report current cycle times of between six and 18 months. Of those, 44% report having to cut their standard development cycles by six to 12 months.

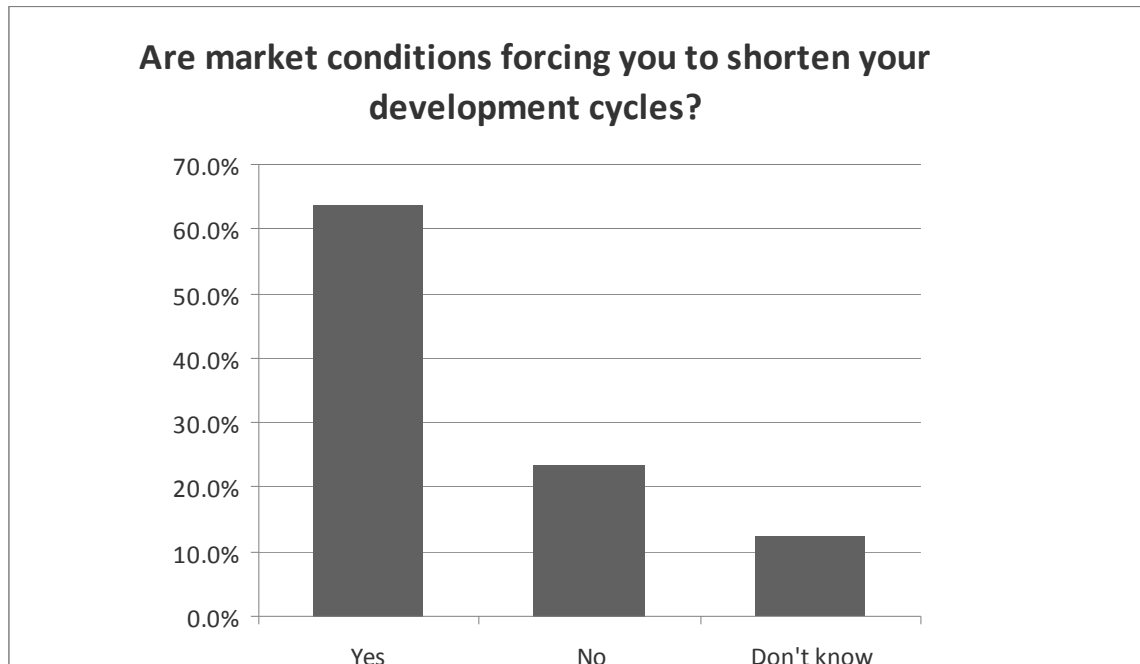


Figure 7: Nearly 65% of respondents are forced to shorten development cycles

The survey also asked about the adoption of new agile development methodologies. The majority (64%) reported that they have not adopted agile methods. However, 57% reported that they are running biweekly build cycles or fewer, with 16% doing nightly builds.

So despite most teams not considering themselves formally agile, most teams are using iterative processes. Surprisingly, 18% indicated that they don't operate on a formal, fixed schedule for their build cycles.

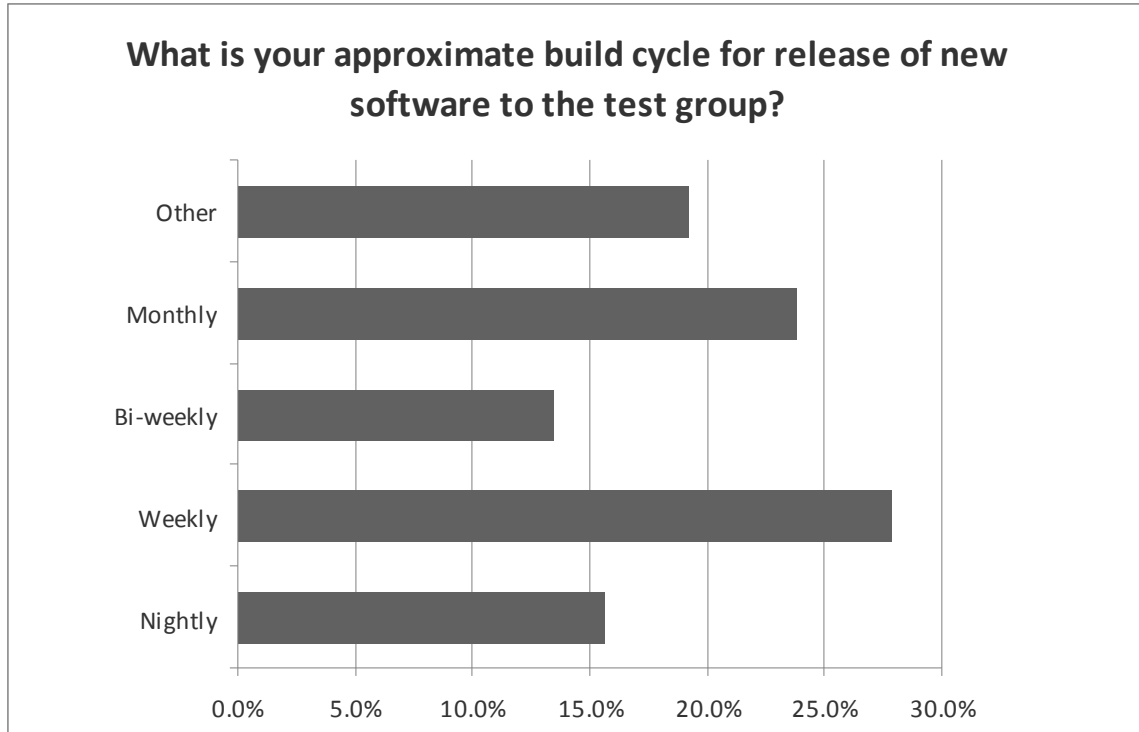


Figure 8: Length of build cycles

A majority of participants (more than 81%) reported that their teams test each new software build on the embedded device itself, showing they recognize the importance of it, rather than just conducting unit tests. Surprisingly, 81% of respondents cited manual testing tools as part of their testing process. Presumably, manual testing within this iterative environment is becoming a challenge.

More than 62% reported that it takes teams a full day or longer to isolate and repair defects found during system integration. More than 25% said it takes them more than two days. This is clearly a bottleneck to the process and an expense that needs to be minimized.

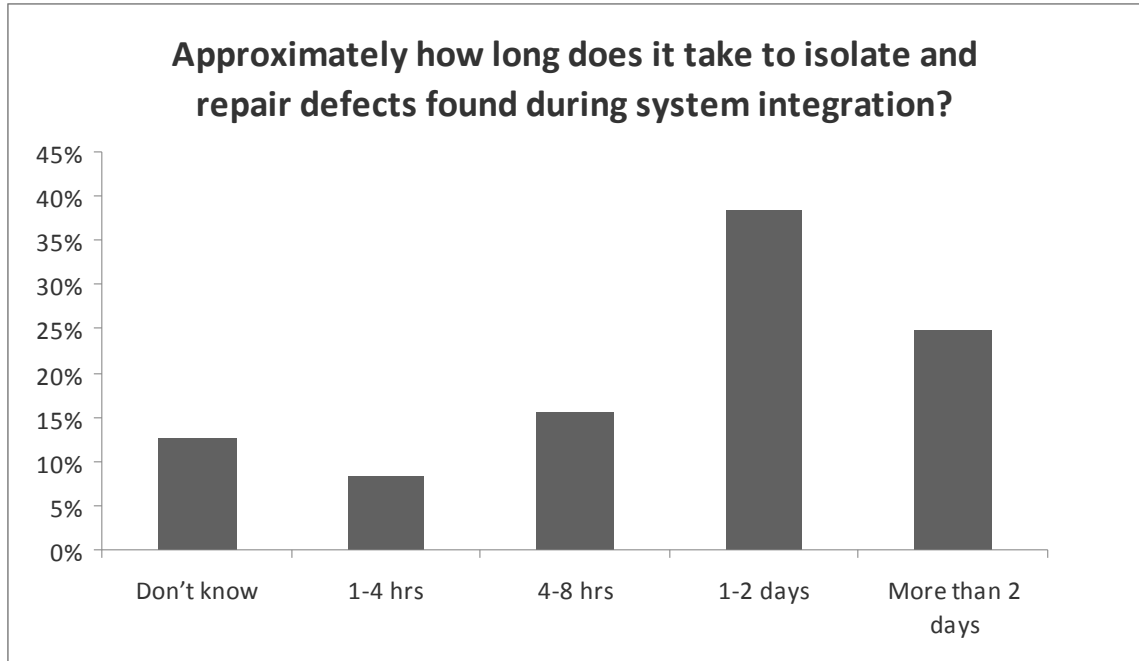


Figure 9: Defect isolation and repair often takes days during system integration

The survey asked how they tested failure modes, error conditions, and performance. Manual methods to verify failure modes were the most common, at 57%. And 8.3% said they did not verify these conditions at all. Of those who knew, 41% said their tests were not able to measure performance of their devices at run-time.

Numerous participants also commented on the difficulty of understanding precisely what is happening within their device under test at run-time.

“Multi-core products are creating performance challenges for the architects and more corner cases, which are difficult to test.”

“Software must be tested thoroughly under all possible operating conditions.”

“(It’s) difficult to ‘see inside’ linked programs.”

In a somewhat surprising finding about their test environments, more than half of all participants (56.9%) reported that their software development and testing processes are not governed by safety, security, or other certification standards. Of those that are, adherence was fragmented across many different standards.

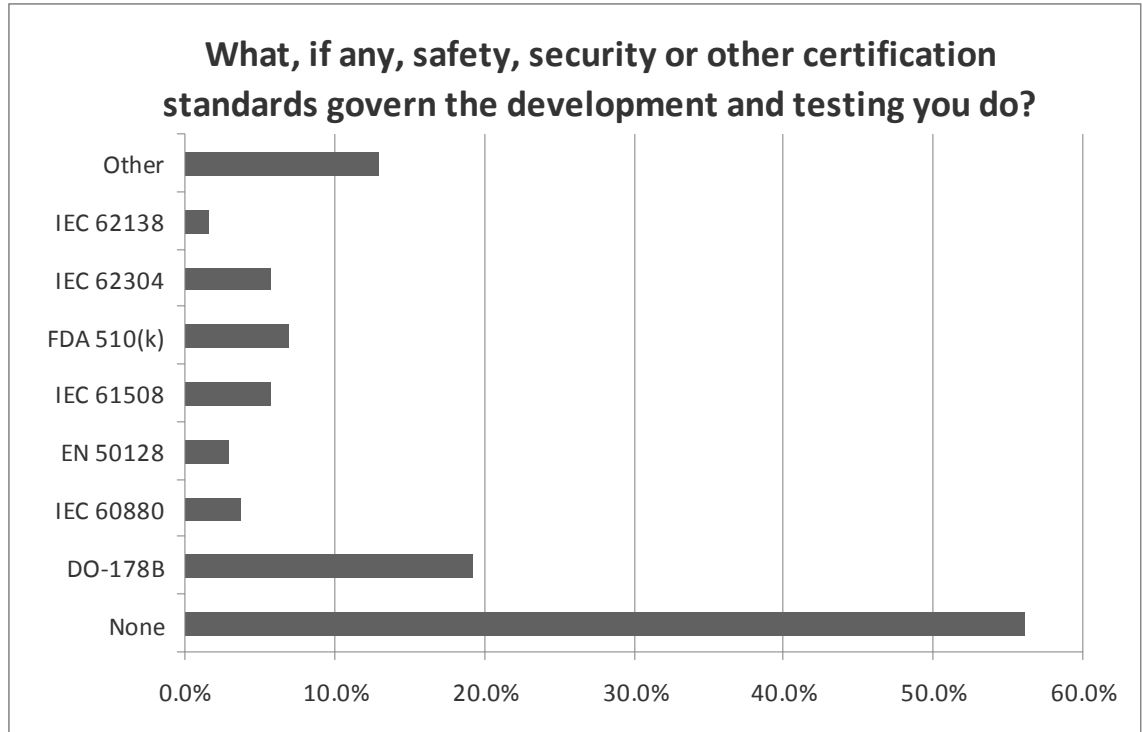


Figure 10: Numerous standards governing development and testing

Inadequate Visibility into Quality

When asked if their existing tools give management sufficient software quality information to make release readiness decisions with confidence, 75% of respondents said they were at best only “somewhat confident.” For 19.1% of respondents, their answer was “No, not confident.”

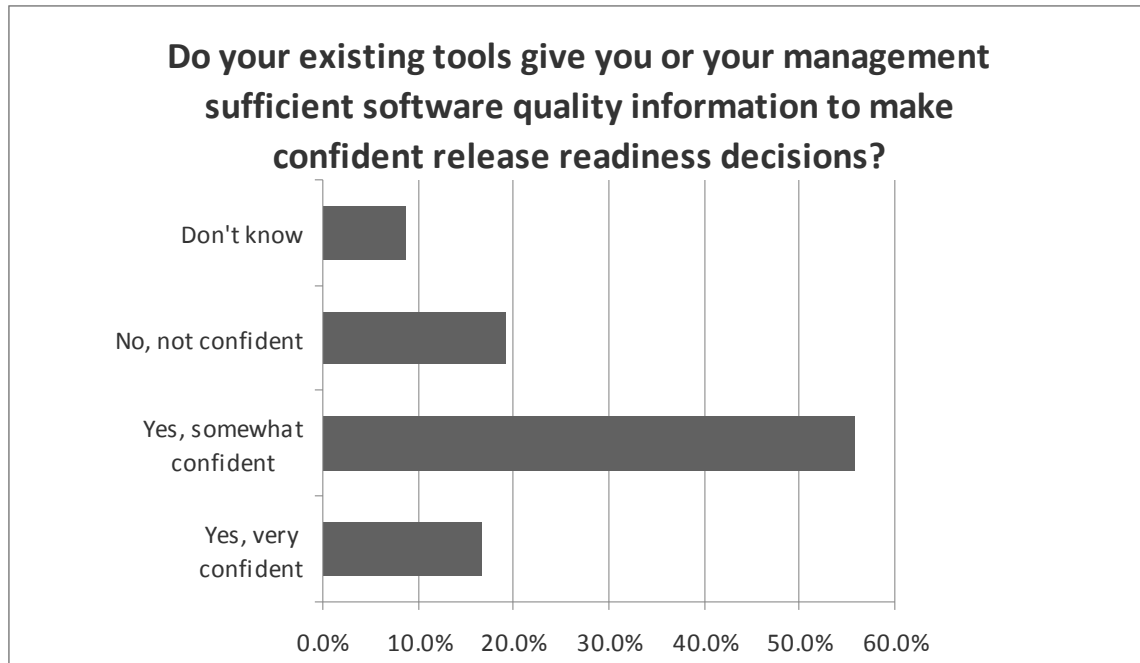


Figure 11: Insufficient quality information is leading to low management confidence

It seems that a high percentage of participants has learned to be skeptical about the software quality information upon which they must make decisions. Of those who responded, 57% said they made release readiness decisions based on inaccurate or insufficient software quality information.

The two most common answers to a question about which metrics respondents use to measure the software quality in their products were the following:

- 1) The tracking of the number of open defects
- 2) The numbers of defects reported by customers

Notably, both of these measures are reactive; they essentially gauge the number of defects already introduced into their products. The other less-used metrics, including test coverage metrics, numbers of tests executed, and so on, are more proactive measures that can be controlled and managed by the companies.

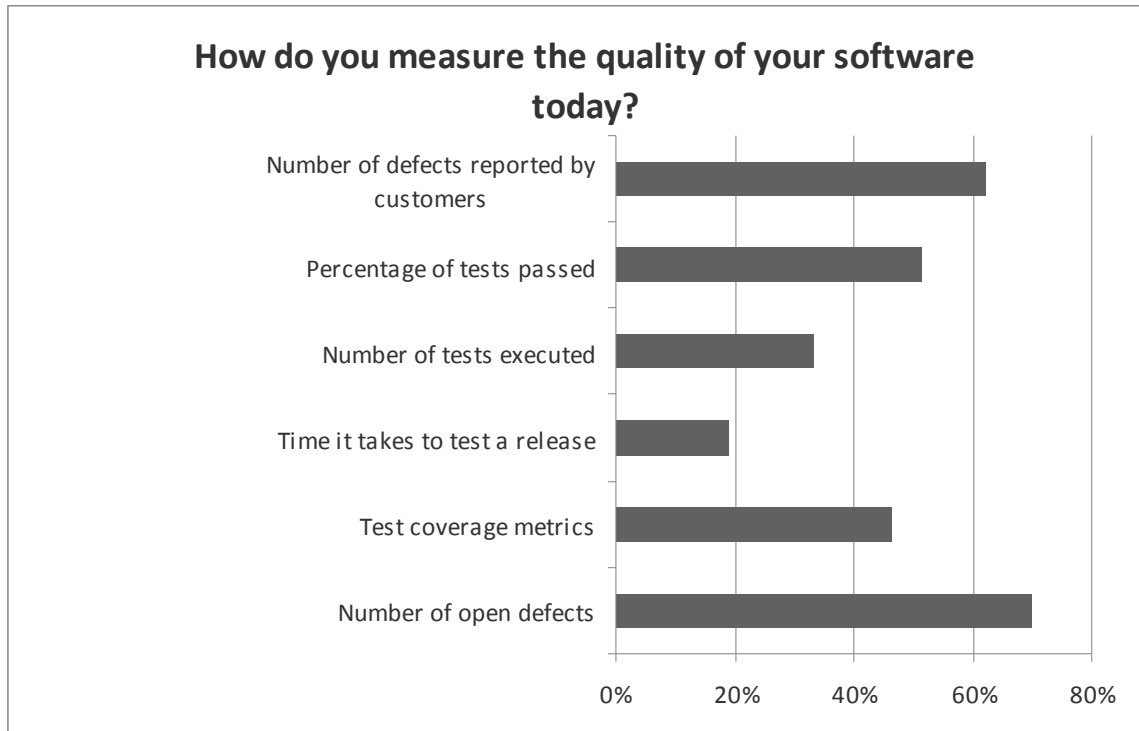


Figure 12: Techniques used to measure software quality

Their lack of confidence and reactive management approach can also be linked to the low use of test coverage and code coverage tools. These tools provide insight into whether software was actually tested and how thorough test suites are. They should be used in both unit testing at the developer's desktop as well as in the functional system testing in the QA lab.

Overall, only a minority of participants use code coverage and test coverage tools. In a question about whether they track which parts of their code are exercised by their tests, more than 60% said they either don't track test coverage at all or only do so during development unit testing. Only about 32% said they measure and track what code is exercised in system tests of the fully integrated device.

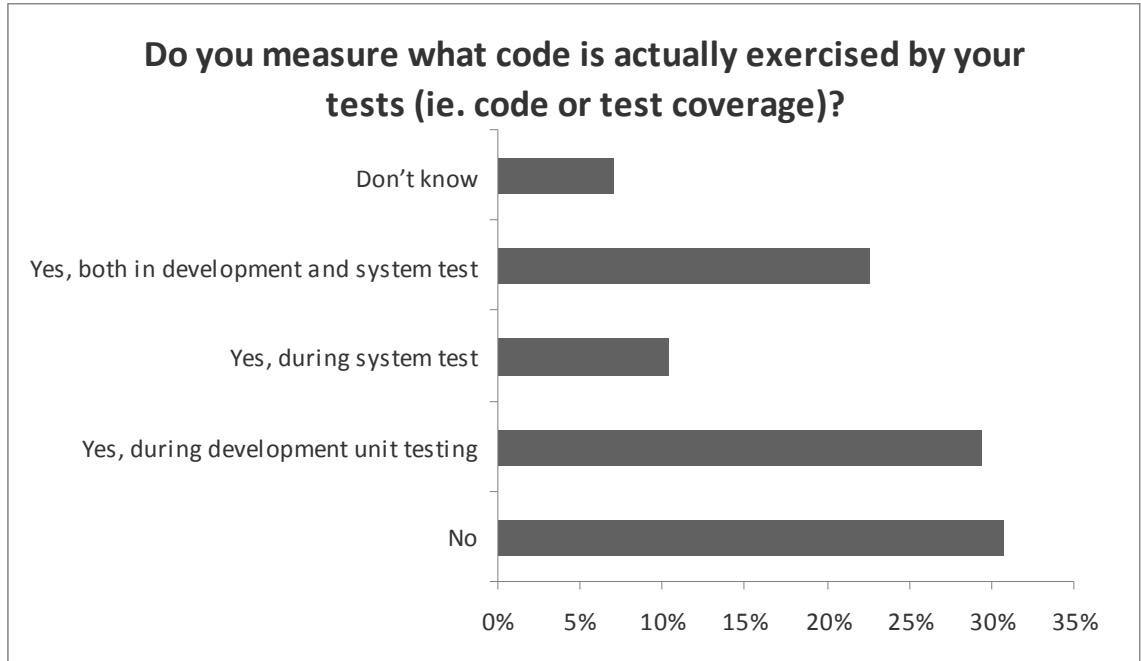


Figure 13: Incomplete measurement of test coverage

In perhaps the most telling response about test coverage goals and actual test coverage results with current projects, nearly 65% of participants reported having a goal of more than 70% for test coverage of their code bases. But only 35% of respondents say they are at greater than 70% code coverage with their current projects.

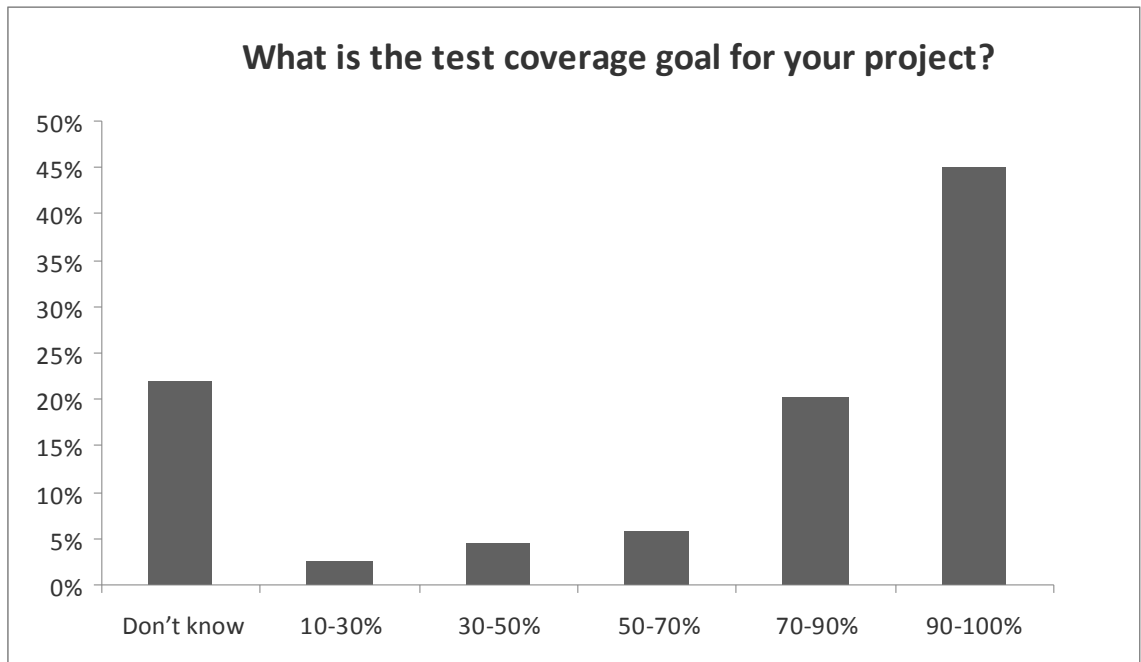


Figure 14: Ambitious test coverage goals

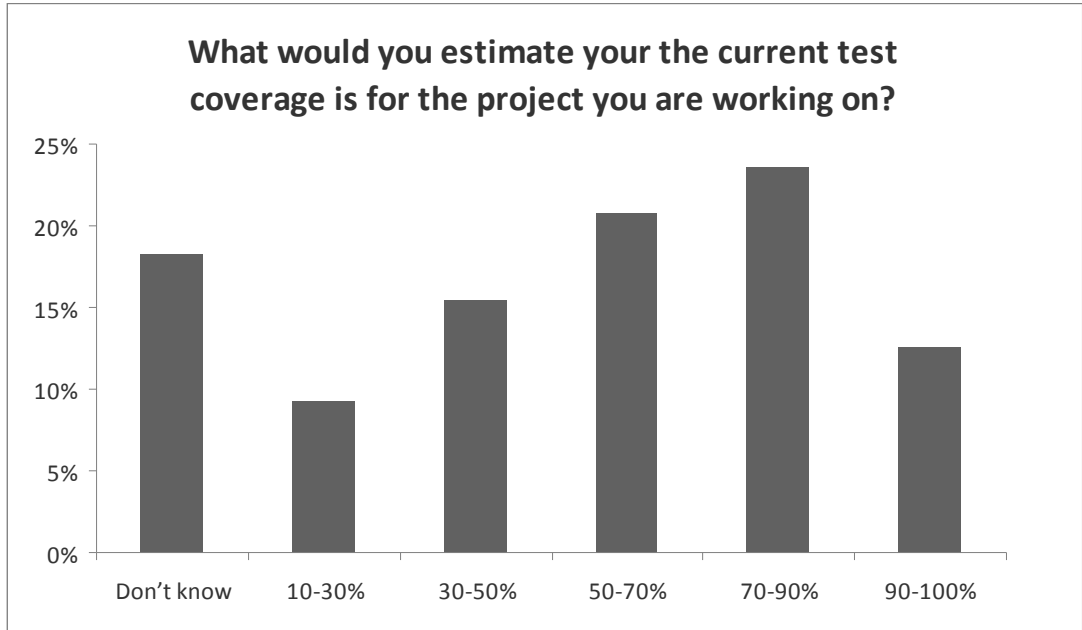


Figure 15: Reality is falling far short of goals for test coverage

High Cost of Poor Quality

Participants were asked how much of their total project resources are spent on testing. More than 77% of respondents reported that testing activities typically consume more than 20% of their total project budgets. Nearly 21% reported testing costs totaling more than 40% of their total project budgets.

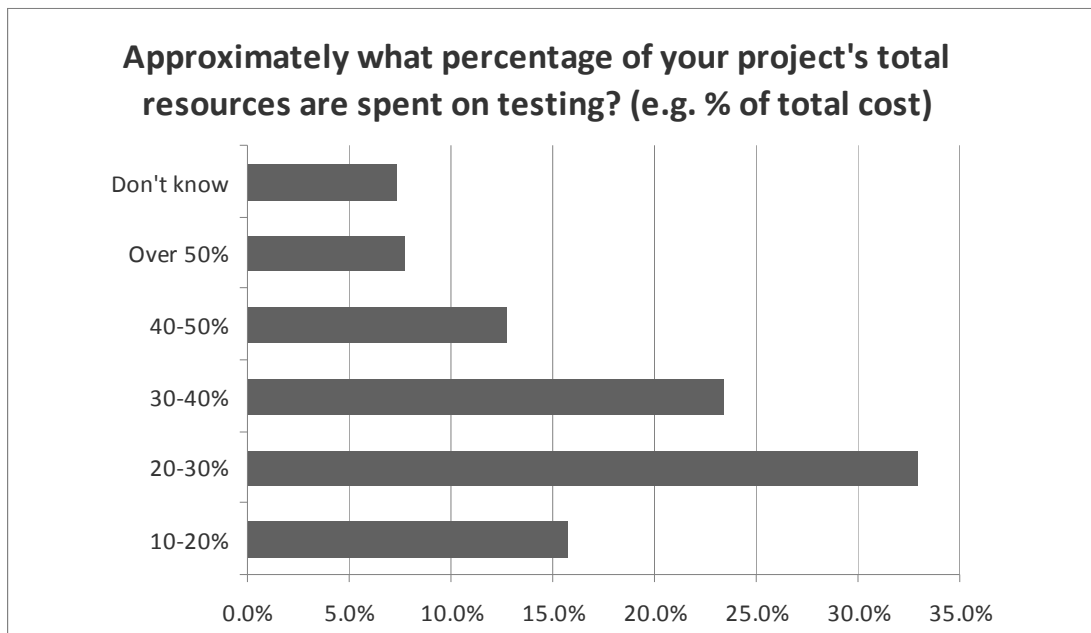


Figure 16: A substantial portion of development project resources is devoted to testing

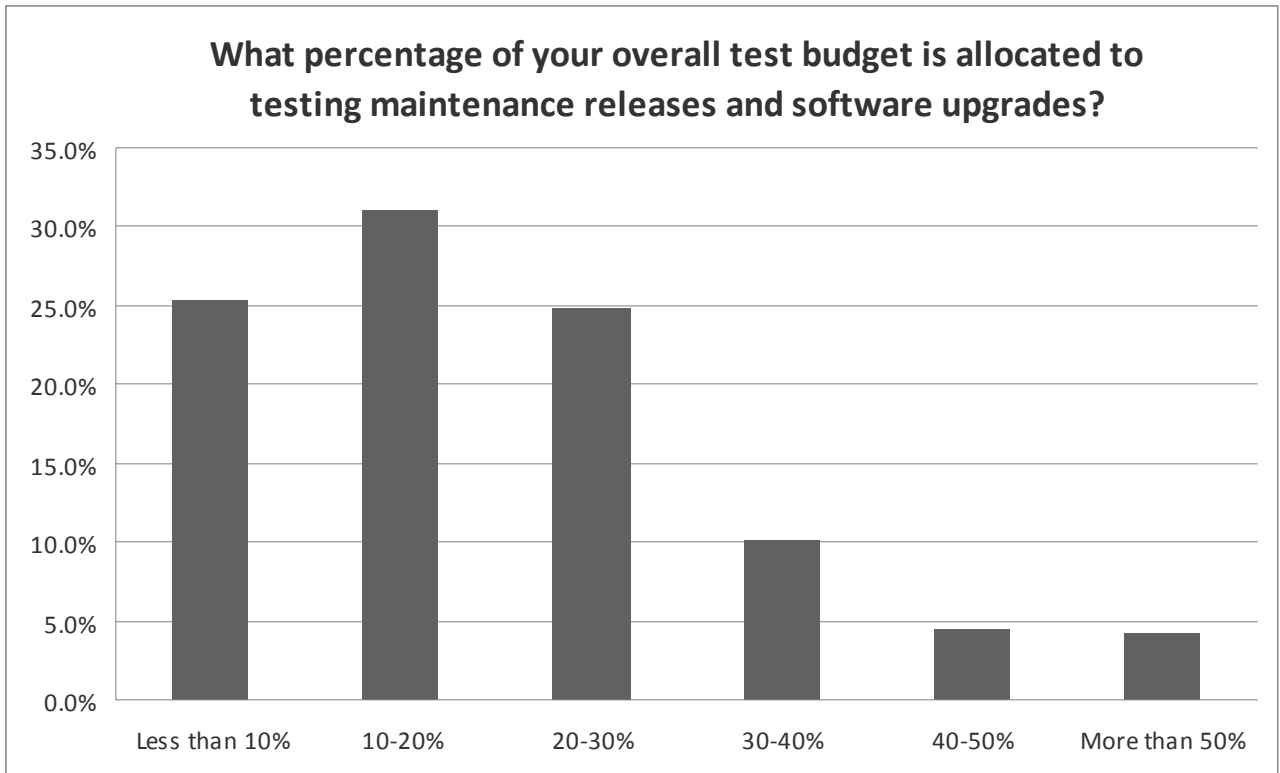


Figure 17: Significant percentage of test budgets for maintenance releases and upgrades

Participants were asked what percentage of their overall test budget is allocated for testing maintenance releases and software upgrades. A majority (74.8%) said they dedicate more than 10% to this lower-profile but still evidently important testing activity.

While 20%–40% of project budget for testing is significant, it is only a small part of the cost of quality, or rather the cost of poor quality.

Participants were asked how poor quality was affecting their development cycle and how they responded to and measured this effect. Over the last year, their projects have been disrupted due to late-cycle discovery of critical software quality problems, according to 72% of respondents. This frequency is surprisingly high, with most respondents reporting that it happened more than once per project in the last year.

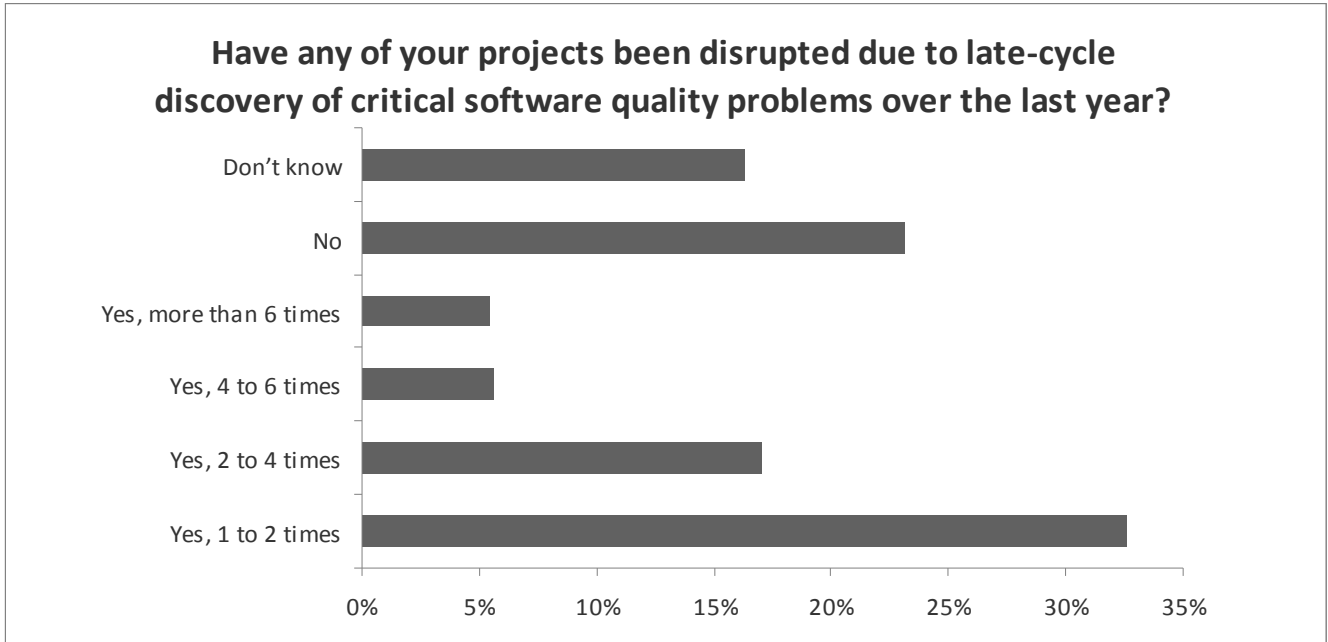


Figure 18: Majority of respondents had projects disrupted by late discovery of defects

Of those who experienced disruptions, 57.6% said they delayed their products' ship dates in order to fix the problems and retest the products.

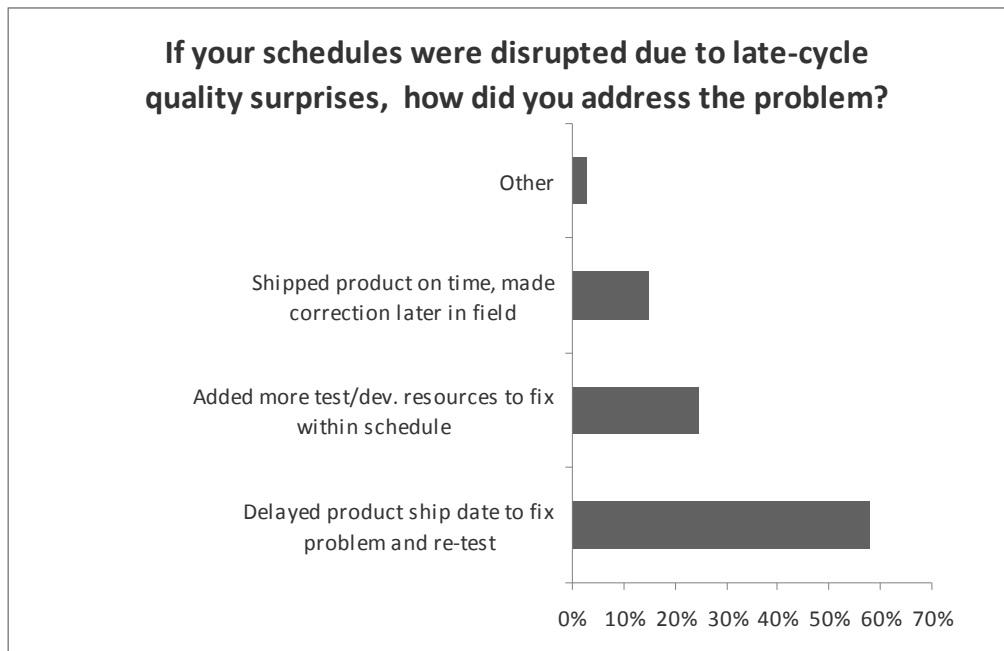


Figure 19: Product delays most often result from late-cycle disruptions

When they did ship products with software defects that were discovered later by customers, it was most often unusual and unanticipated use that exposed the problems:

- 1) Products used in unanticipated ways (58.1%)
- 2) Untested edge conditions (48.9%)

When you do ship products with software defects, generally what have you found to be the source of the problems?

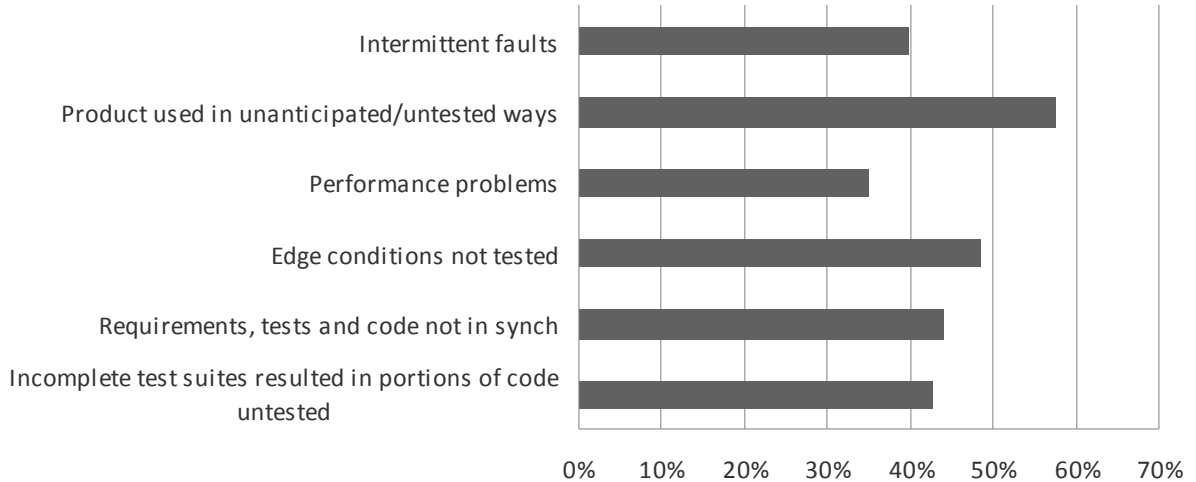


Figure 20: Untested uses are largest source of delivered software defects

When asked how they measured the cost to the company when products were not delivered at planned quality, they reported the following as the top three business drivers:

- 1) Higher than expected program costs
- 2) Damage to brand due to shipped defects
- 3) Missed market window for the product

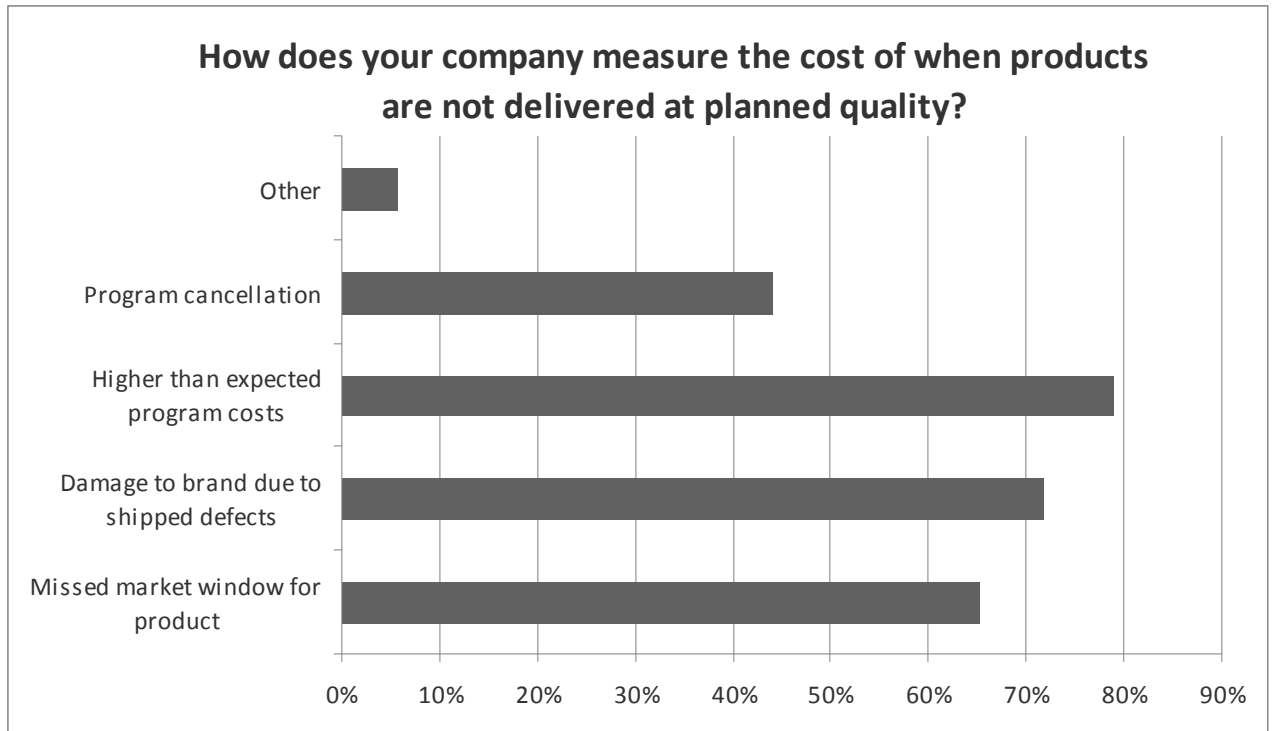


Figure 21: Delivered defects damage brand and limit market window

Investing in Test Automation

Industry trends and market forces have pushed the requirements for software quality testing well beyond the capabilities of the tools and processes currently in use. The problem is getting worse with more code crammed into devices, off-the-charts architectural complexity, and squeezed development and test cycles. This is the new normal.

This disparity is causing serious operational problems for embedded products companies, negatively impacting not only their brands but also their bottom-line results. Ignoring the problem is no longer an option. Leaders are seeking new test automation tools that can help them with the unique problems of embedded device testing while helping them deliver on time, on budget, and on quality.

One respondent summarized it best: “We spend too little time testing and too much time hoping.”

Participants were asked various questions about their present software quality-testing environment and their plans and priorities for new investments to improve their testing operations in the coming year.

As for the tools currently in use in their testing processes today, respondents not surprisingly reported that they use a wide variety of systems and techniques. Results show that manual testing, conducted by 81.7% of respondents, remains a substantial part of the device test processes at most organizations today. Other well-represented categories include defect tracking systems, static or dynamic code analysis, and performance and timing measurement.

There was a strong showing by simulators, which were reported as being used by more than 50% of survey participants. This category includes both device simulation environments as well as traffic, load, and other external simulators that drive device behavior during the test process.

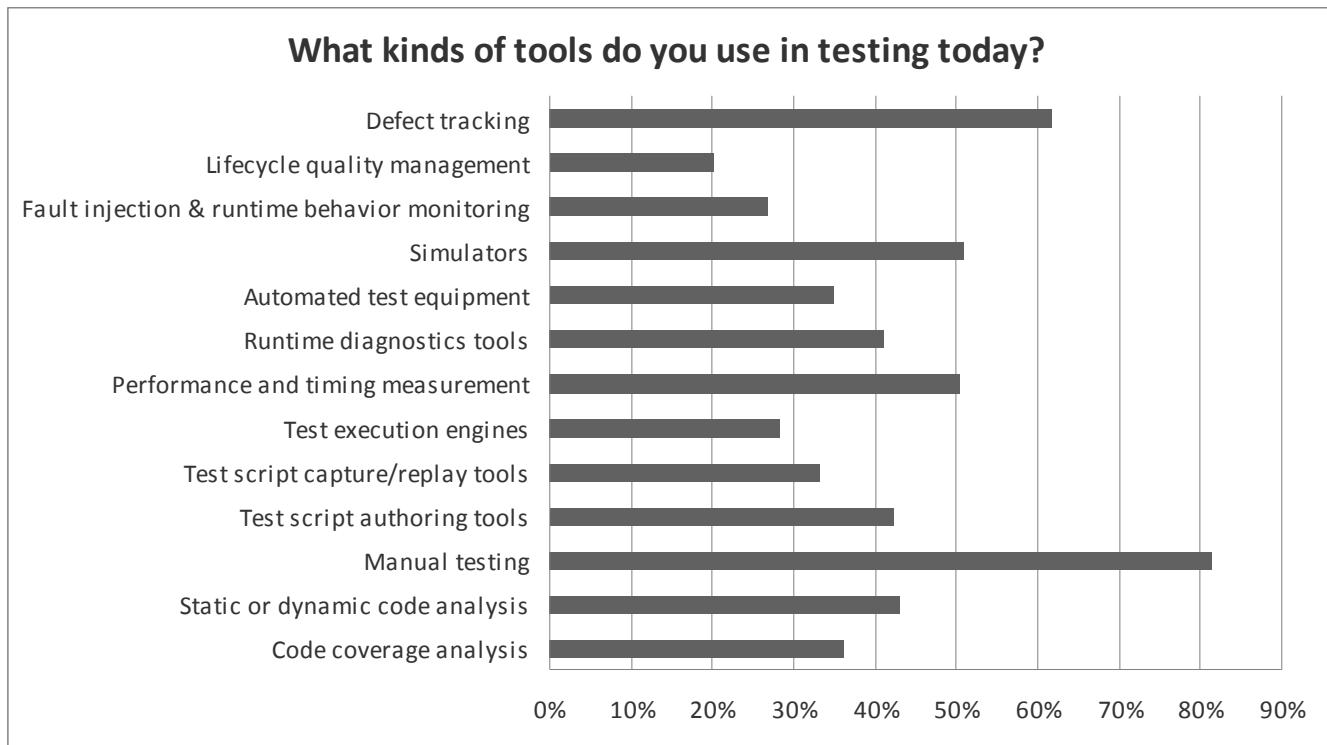


Figure 22: A wide variety of software testing tools in use today

Regarding plans for new investments, there is a clear need for more automated testing among respondents. The top 10 tools for future investment are the following:

- 1) Automated test equipment
- 2) Code and test coverage analysis
- 3) Simulators
- 4) Run-time diagnostics tools
- 5) Static or dynamic code analysis
- 6) Performance and timing measurement
- 7) Test execution engines
- 8) Script authoring tools
- 9) Test script capture/replay tools
- 10) Fault injection and run-time behavior monitoring

At least five of these tools are focused on helping testers more effectively understand what is happening in their code at run-time. Respondents indicated that purchasing tools for these run-time analytics is a higher priority than obtaining more traditional tools for manual testing, defect tracking, or life cycle quality management.

The survey also asked an open-ended question about other trends or investments that should be considered. A significant number of respondents expressed the need for better host-based testing and device simulation to streamline testing early in the process before the hardware is available and to reduce the cost of testing assets.

“The biggest problem we have is we are not able to test for a new ASIC before the ASIC is taped out ... It would be nice if there were tools that could be used to develop production code alongside the simulator that we can use for testing before an ASIC arrives and run them in the high-level system.”

“There is a strong need for a host-based ‘device dependent code’ testing suite or methodology.”

“We need better simulations of embedded devices because real hardware can be expensive.”

Survey Statistics Summary

- 54% are experiencing growth in software content by 25%–100% annually.
- 96% are updating software in fielded devices.
- 80% are testing every build on the production devices.
- 64% report shortened development cycles due to market conditions.
- 44% of those with reduced schedules report life cycle reductions of 6–18 months.
- 57% have moved to iterative build cycles of two weeks or less.
- 36% are using agile or similar methods.
- 62% take one day or longer to isolate and repair each defect.
- 81% are using manual testing methods.
- 41% are not able to validate run-time performance.
- 65% have goals to achieve greater than 70% test coverage of their code base.
- 60% do not measure code or test coverage.
- 35% are hitting the goal of more than 70% coverage.
- 32% look at code coverage of the fully integrated device.
- 15% shipped products to customers with a critical bug included (with the intent of fixing it later in the field).
- 13% feel very confident on the thoroughness of their tests.
- 77% spend more than 20% of their total project budgets on testing activities.
- 21% report testing costs totaling more than 40% of their total project budgets.
- 57% have made release readiness decisions based on inaccurate or insufficient software quality information.
- 72% have had disrupted projects over the last year due to late-cycle discovery of critical software quality problems.
- 57% of those that experienced quality disruptions said they delayed their products' ship dates in order to fix the problems and retest the products.
- 58% of the time, products used in unanticipated ways is the cause of defects delivered to customers, and 49% of the time the cause is untested edge conditions.
- More than 50% use simulators as part of testing.
- 68% plan to automate more of their test processes in 2010.
- 70% track open defects, and 62% track customer-reported defects as main quality metrics.
- 44% make product readiness decisions on information that turned out to be inaccurate.

About Wind River

Wind River, a wholly owned subsidiary of Intel Corporation (NASDAQ: INTC), is a world leader in embedded and mobile software. Wind River has been pioneering computing inside embedded devices since 1981, and its technology is found in more than 500 million products. Wind River is headquartered in Alameda, Calif., with offices in more than 15 countries.

Wind River makes a variety of tools and solutions for device development and testing, including the Wind River Test Management test automation system for monitoring, executing, and managing embedded device software testing. Wind River Test Management delivers insights at run-time, enabling teams to make informed decisions that improve software quality and mitigate risk during the testing process. Wind River Test Management lets teams optimally execute complex tests while dynamically gathering information from the production software under test as it is running, without requiring special pre-instrumented software builds. This approach allows teams to adopt new white-box test techniques that give testers visibility into the operation of the device and helps them determine the thoroughness of the tests, quickly identify defects and performance bottlenecks, and focus efforts on sections of software that are most in need of testing.

To learn more, visit Wind River at www.windriver.com/products/test_management and blogs.windriver.com.

Wind River is a world leader in embedded and mobile software. We enable companies to develop, run, and manage device software faster, better, at lower cost, and more reliably. www.windriver.com

© 2010 Wind River Systems, Inc. The Wind River logo is a trademark of Wind River Systems, Inc., and Wind River and VxWorks are registered trademarks of Wind River Systems, Inc.