2011 Model-based Testing User Survey: Results and Analysis



Robert V. Binder

Copyright © 2012, Robert V. Binder. All Rights Reserved

2011 Model-based Testing User Survey: Results and Analysis

Contents

Overview 4
Survey Highlights 4
Who is a Typical MBT User? 5
Approach 6
Call for Participation 6
Data Collection 6
Definitions 6
Representativeness 7
Survey Responses 8
Analysis and Presentation 8
User Organizations 9
What kind of organization sponsored the model-based testing project? 9
Approximately how many persons work in this organization, in total? 9
Environment and Applications 10
What was the overall process approach used for the MBT project? 10
What is the general application domain of the SUT? 10
What was the general release level of the SUT after testing was completed? 11
How is the SUT provided to end users/customers? 11
What kinds of computers are used in the SUT? 12
What is the general architecture of the SUT? 12
What OS(s) are used in the SUT? 13
In what programming language(s) is the SUT implemented? 14
Impetus and Preparation 15
What factors led to your project using MBT? 15
At what stage of MBT adoption is your organization? 16
MBT Usage and Configuration 18
What kind of MBT tools were used in the project? 18
If you used a commercial tool, which one? 18
If you used an open source MBT tool(s), please indicate which. 19
If you used an MBT tool(s) developed for use in your organization, please describe briefly. 19

Does your MBT tool have useful integration (get/save data) with other development environment systems? 20
To what extent is MBT testing integrated into your production build cycle? 22
Points of Control and Observation 22
What is the general approach for connecting MBT tool output, SUT control interfaces, and SUT observation interfaces? 22
What kind of SUT interfaces are controlled to setup and/or execute MBT test cases? 23
What kind of SUT interfaces are observed to provide test responses for automated MBT evaluation? 23
Testing Practices 24
Which aspects of the SUT do you test with MBT? 24
At what scope do you use MBT? 24
What kind of coverage is measured for your MBT test runs? 25
Time, Cost, Quality, and Effectiveness 26
What has your experience been with some common MBT problems? 26
What is your best estimate of the percent increase or decrease in escaped bugs after testing with MBT? 27
What is your best estimate of the percent increase or decrease in testing costs with MBT? 27
What is your best estimate of the percent increase or decrease in testing duration with MBT? 27
In your opinion, how do others view the overall effectiveness of MBT? 28
Overall, how effective do you think MBT has been? 28
Future Use 30
How likely are you to continue to use MBT? 30
Describe the biggest limitation of MBT. 30
Describe the biggest advantage of MBT. 31
Additional comments about this survey or MBT. 31
Selected Publications 32
About System Verification Associates 33

Figures

Figure 1 Definitions for Survey Terms 6
Figure 2 User Organization Sector 9
Figure 3 User Organization Size 9
Figure 4 User Software Process 10
Figure 5 SUT Domain 10
Figure 6 SUT Release Level 11
Figure 7 SUT Business Model 11
Figure 8 SUT Platforms 12
Figure 9 SUT Architecture 12
Figure 10 SUT Operating Systems 13
Figure 11 SUT Programming Languages 14
Figure 12 Impetus for using MBT 15
Figure 13 Stage of Adoption 16
Figure 14 MBT Skill Development Approach 16
Figure 15 Average Training Time to Proficiency 17
Figure 16 MBT Provider Types 18
Figure 17 Commercial MBT Product Used 18
Figure 18 Open Source MBT Systems Used 19
Figure 19 Proprietary MBT Systems Used 19
Figure 20 MBT/Environment Integration 20
Figure 21 Proportion of Testing Modes 21
Figure 22 Average and Median Allocation by Testing Mode 21
Figure 23 MBT Build Integration 22
Figure 24 MBT/Test Harness Integration 22
Figure 25 MBT Points of Control 23
Figure 26 MBT Points of Observation 23
Figure 27 Aspects Tested with MBT 24
Figure 28 Scope of Testing with MBT 24
Figure 29 Test Coverage Measured 25
Figure 30 Impact of Common Problems 26
Figure 31 Effect of MBT on Escaped Bugs 27
Figure 32 Effect of MBT on Testing Costs 27
Figure 33 Effect of MBT on Testing Duration 27
Figure 34 Views of others on MBT Effectiveness 28
Figure 35 Overall Effectiveness of MBT 28
Figure 36 Planned Future Use 30
Figure 37 Statements about MBT Limitations 30
Figure 38 Statements of MBT Advantages 31
Figure 39 General Statements about MBT 31

Overview

Survey Highlights

The industrial use of model-based testing (MBT) started a little over 15 years ago with the introduction of the Test Master product.[1] Put simply, model-based testing is the use of a software system to represent aspects of a system under test (a model) to generate test cases. Please refer to *Selected Publications* for additional information.

I conducted this survey to establish a profile of present-day users of model-based testing.

- Provide facts about who, what, when, why, and how.
- Check previously reported obstacles.
- Check benefits and costs.
- Check assumptions about success/fail factors.
- Check how MBT and software process aspects are associated.

The responses indicate that model-based testing is a practical and effective software development technology.

- MBT usage spans a wide range of application stacks, software processes, application domains, and development organizations. These include open source and commercial providers; embedded, transaction processing, and communications applications; C/C++, Java, C#, and other programming languages; and process models including none, Agile, incremental, and V-model.
- MBT is accessible and practical: half of the respondents report becoming proficient MBT with 80 or fewer hours of training or coaching, 80% with 100 or less.
- A wide range of tool chains are in use from commercial, open source, and proprietary providers.
- On average, respondents report MBT reduced escaped bugs by 59%.
- On average, respondents report MBT reduced testing costs by 17%.
- On average, respondents report MBT reduced testing duration by 25%.
- Four in five users do not see the problems reported by MBT early adopters.
- To the extent that other software process stakeholders are aware of MBT, they view it as either effective or neutral.
- Seventy-nine percent of the respondents are very or extremely likely to continue using MBT.

Several respondents noted common challenges.

- Despite advances in tools, the intellectual task of modeling and devising effective physical abstractions for SUT interfaces can be difficult.
- A tool alone cannot produce models, so when a system is poorly understood or documented, MBT users work at a disadvantage and may be ineffective. Some sponsors apparently believe MBT is magic pixie dust.

[1] Larry Apfelbaum, "Spec-based tests make sure telecom software works," *IEEE Spectrum*, November 1997, 77-83

Who is a Typical MBT User?

Taking the most frequent responses for all users, here is how a hypothetical composite MBT user would describe their situation and usage.

I work for a large business organization. We follow an agile process to develop the software we test. The system under test is embedded in another product sold to our customers. It is programmed in C++, runs on dedicated computers, is networked, and uses a Windows OS.

We felt MBT could help reduce testing cost/work and bug escapes. We use a commercial MBT tool that we've integrated with our design repository and test harness.

We've added MBT to the mix of our existing manual testing and traditionally programmed test drivers. Our effort for each testing mode is roughly equal.

At present, we're using MBT in a pilot project to test version 1.0 of our product. We taught ourselves to develop test models and use the tool by reading documentation and experimenting. It took us each about 100 hours of this self-study to become minimally proficient with the tool.

We use MBT to test system-scope functionality. The tool generates code and data for an adapter that abstracts the SUT APIs and then drives the APIs and evaluates results from the APIs. We measure requirements coverage for our test runs. The output of this tool is not integrated with our build.

We've had some difficulty developing test models. Sometimes our model "blows up" (combinatorial runaway) and we've found that our MBT test suites are not effective for certain kinds of bugs. However, this is manageable and not any worse than what we expected. With respect to other first generation MBT problems (updating, integration, and inadequate oracle), either we have not seen these problems or they're not obstacles.

In all, we think MBT has been moderately effective so far. For the most part, our expectations for both challenges and improvement have been met. MBT has reduced bug escapes about 60%, testing costs by 15%, and testing time by 30%. Among our co-workers, developers and managers are neutral about MBT but other testers view it as effective. Our users/customers are not aware of MBT. Overall, we are very likely to continue using MBT.

Approach

Call for Participation

The survey was announced to the attendees of the first Model-based Testing User conference, held October 18-20, 2011 in Berlin.[2] Calls for participation were also posted on MBTUC blog site and in public online forums: Linkedin.com Model-based Testing user group, Spec Explorer Forum, IBM Rational forum, the UseNet group comp.software.testing, and SQA Tester forum. The call was also emailed to several dozen participants of Advances in Model Based Testing (AMOST) workshops and others with an interest in MBT.

Data Collection

The survey was conducted online using the Survey Monkey web site. The following instructions were presented to respondents.

> The purpose of this survey is to learn about how model based testing is being used. We ask for some basic environment and organization information and about how MBT is used and viewed. The questions are designed to find what broad usage patterns exist. All replies are strictly confidential. If you would like to receive a copy of the survey report, please provide your email address on the last page. It should take about 15 to 30 minutes to complete this survey. Your cooperation will support MBT advocacy and provide useful feedback for all tool providers.

The survey contained 35 questions, which are all listed in the next section.

Definitions

The survey questionnaire provided the following definitions.

Bug	Defect, issue, trouble report, fault, failure, etc.
Model-based Testing (MBT)	The use of a software system that represents abstract aspects of a system under test to generate test cases.
Tool	An MBT software system/program.
System Under Test (SUT)	A software system that has been tested with model based testing.
Project	A project in which you have used MBT.
Oracle	A trusted source of expected test results (not the database).
Organization	The business, agency, school, department, etc. which sponsored the project.

Figure 1 Definitions for Survey Terms

[2] http://www.model-based-testing.de/mbtuc11/index.html. This event was organized by the European Telecommunications Standards Institute (ETSI) and the Fraunhofer-Institutes FIRST and FOKUS in Berlin. This is the first meeting that focused on industrial application of MBT. Before this, workshops (like AMOST) and conferences about MBT have focused on academic research.

Representativeness

Over a thousand persons probably viewed the call for participation. This produced 52 total responses, suggesting a response rate around 5%. I dropped ten responses that did not indicate any kind of MBT tool and four responses that did not answer any usage questions. I analyzed the remaining 38 complete responses.

This survey was not intended as nor can it be considered broadly representative. Participants self-selected and the canvas efforts were necessarily limited. However, the respondents represent many kinds and sizes of organizations, software products, and software processes. Business organizations were the only category with dominant representation– about 80% of all respondents.

The instances and proportions of tools reported here should not be interpreted as indicative of market share, nor should the presence or absence of any particular tool be interpreted as any kind of endorsement or lack of endorsement.

I did not develop any cross-tabulations of the responses. Owing to the small sample size, meaningful correlation statistics could not be produced. I looked for other patterns in effects like time, cost, and quality and factors like software process or average training time, but did not note any. For example, the average training time for respondents reporting above or below average reduction in bug escapes differs by only three hours. I suspect that with a larger sample, correlations among factors and effects could be found, and that they would be similar to those seen in comparable software technologies.[3] Nonetheless, I believe the data unambiguously show that MBT can make a significant and cost-effective contribution to software quality and productivity.

[3] For example, see Capers Jones and Olivier Bonsignour, *The Economics of Software Quality.* Addison-Wesley Professional, 2011.

Survey Responses

Analysis and Presentation

The survey questions are given in as headers in this section. The respondent answers are graphed or listed, with a brief discussion.

- Most the survey questions presented a list of responses. Some questions had an "other" category, which allowed respondents to enter their own category item. These responses are individually listed. For example, respondents entered all of the single response languages listed in Figure 11 *SUT Programming Languages*.
- Certain questions were open-ended. The responses to these questions are reproduced with some minor editing in Figure 19 *Proprietary MBT Systems Used*, Figure 37 *Statements about MBT Limitations*, Figure 38 *Statements of MBT Advantages*, and Figure 39 General *Statements about MBT*.
- Multiple responses were acceptable for about half of the questions, as in Figure 17 *Commercial MBT Product Used*. The charts for multiple response questions show percentage of all respondents and percent of all responses. For multi-response questions, the sum of all-responses values usually exceeds 100%, as some respondents selected more than one category. For example, in Figure 5 *SUT Domain*, 15 out of 35 respondents (45%) reported developing embedded applications. In total, 59 responses were provided to this question so that embedded applications was 25% of all responses.
- In questions that only accepted a single response, the total number of responses is given for each category.
- A few questions asked for a numerical value (e.g., average training hours per person.) These responses are shown as a distribution, with average and median noted in the discussion.
- Some respondents did not complete all questions. So, although there were 38 individual response sets, the total number of responses for certain questions is less than 38. Percentages are based on the total responses for each question.

User Organizations

Respondent organizations are nearly all are business entities. Large (over 10,000 total persons) and small organizations (under 500) account for 82% of the respondents.

What kind of organization sponsored the model-based testing project?

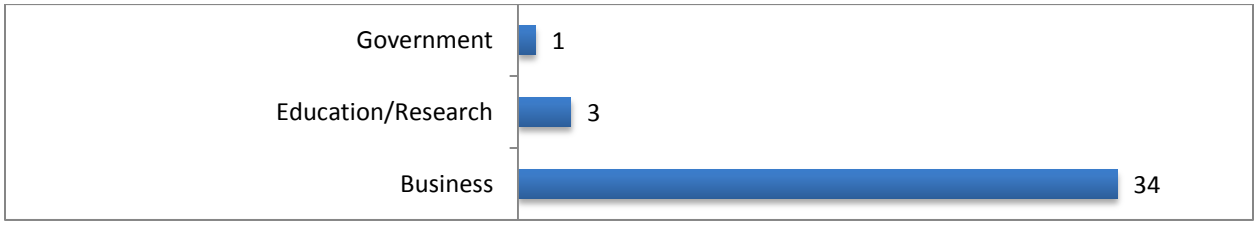


Figure 2 User Organization Sector

Approximately how many persons work in this organization, in total?

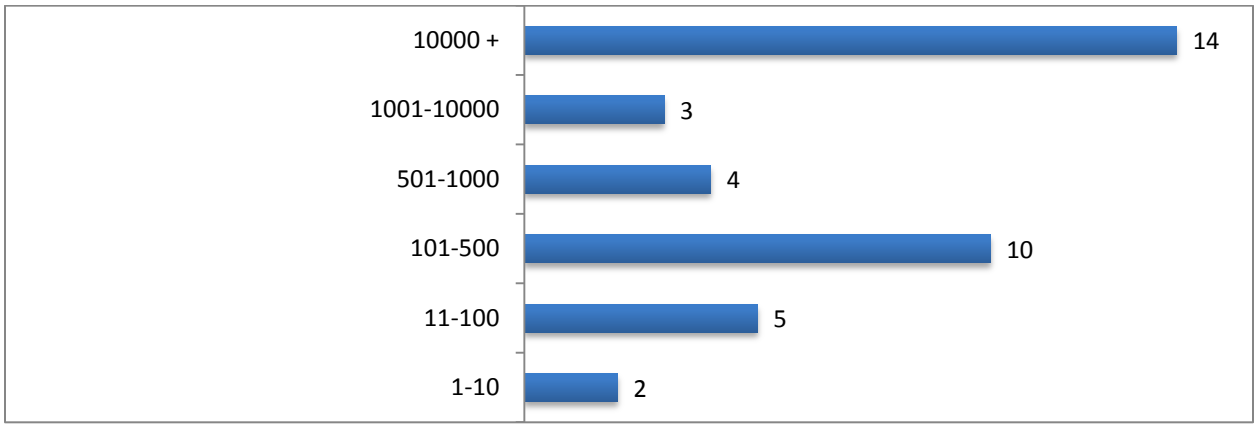


Figure 3 User Organization Size

Environment and Applications

What was the overall process approach used for the MBT project?

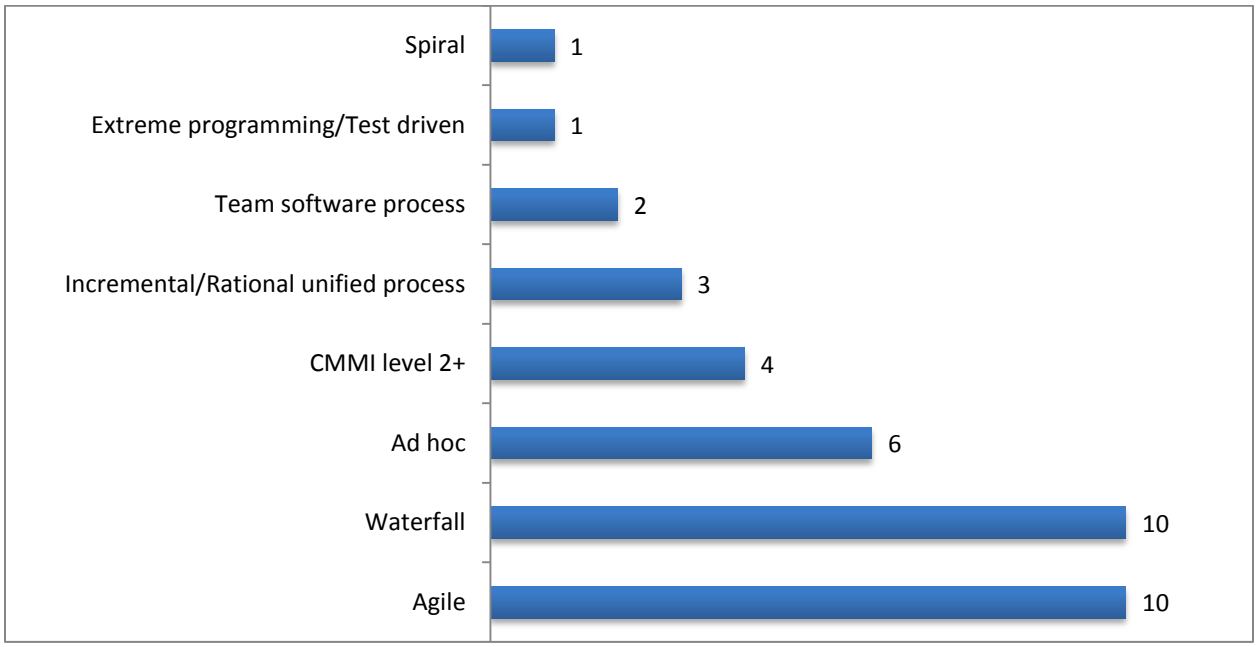


Figure 4 User Software Process

Respondents reported the entire range of software process models. Eighty-four percent use a defined process of some kind; 16% report an ad hoc approach.

What is the general application domain of the SUT?

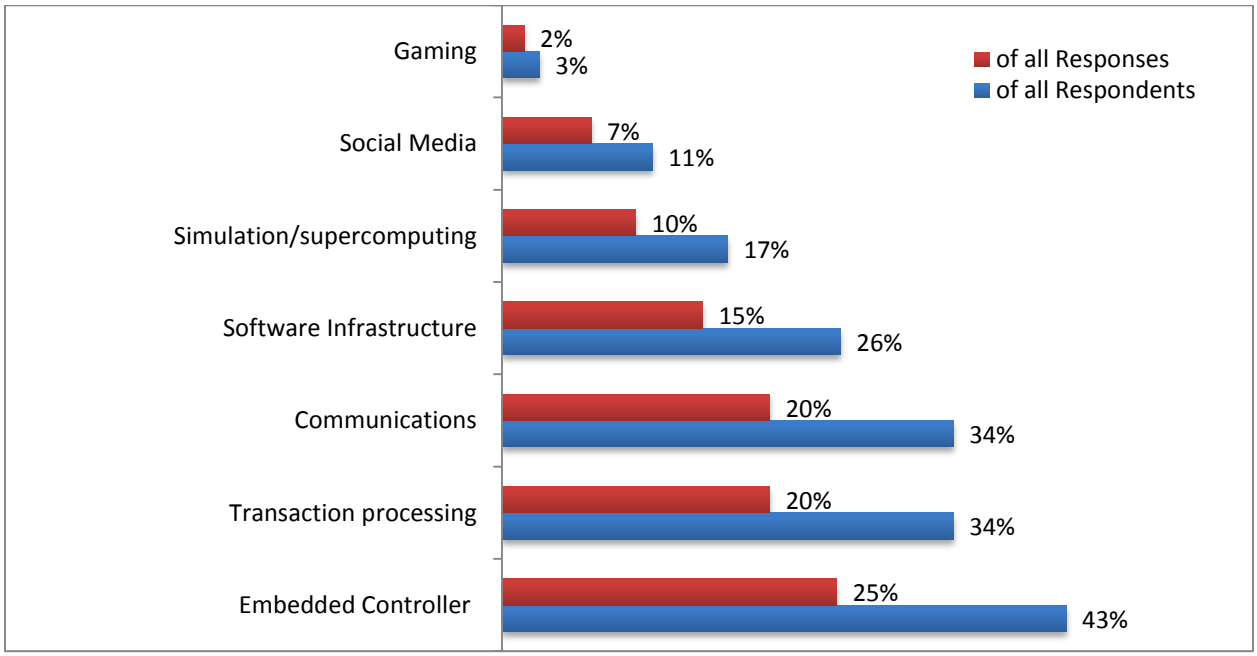


Figure 5 SUT Domain

Survey respondents were primarily developers of embedded applications, transaction processing, or communications systems. Over 90% of the respondents develop for application domains where high reliability is expected (embedded, transaction processing, communications, infrastructure, and simulation.) Of the 35 respondents, 12 (32%) report developing for two or more application domains.

What was the general release level of the SUT after testing was completed?

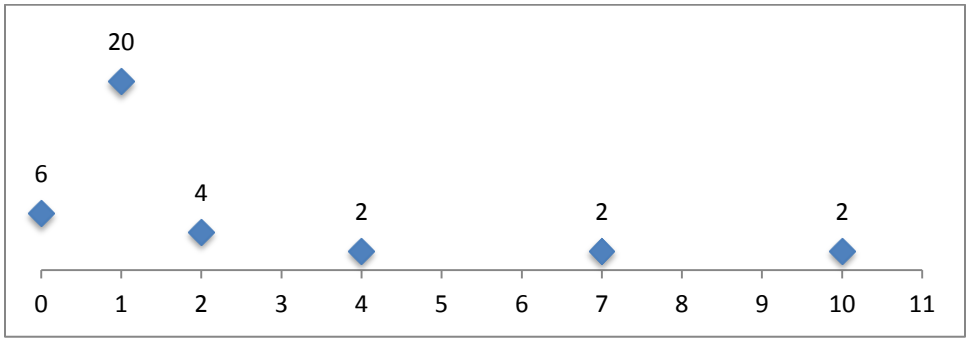


Figure 6 SUT Release Level

Release level is a proxy for the life-cycle stage of the respondent's SUTs, and suggests the extent is MBT used for mature or recently developed systems. Figure 6 shows the number of respondents by major release level of the SUT. Seventy-nine percent of respondents use MBT for newer (version 0, 1, or 2) codebases.

How is the SUT provided to end users/customers?

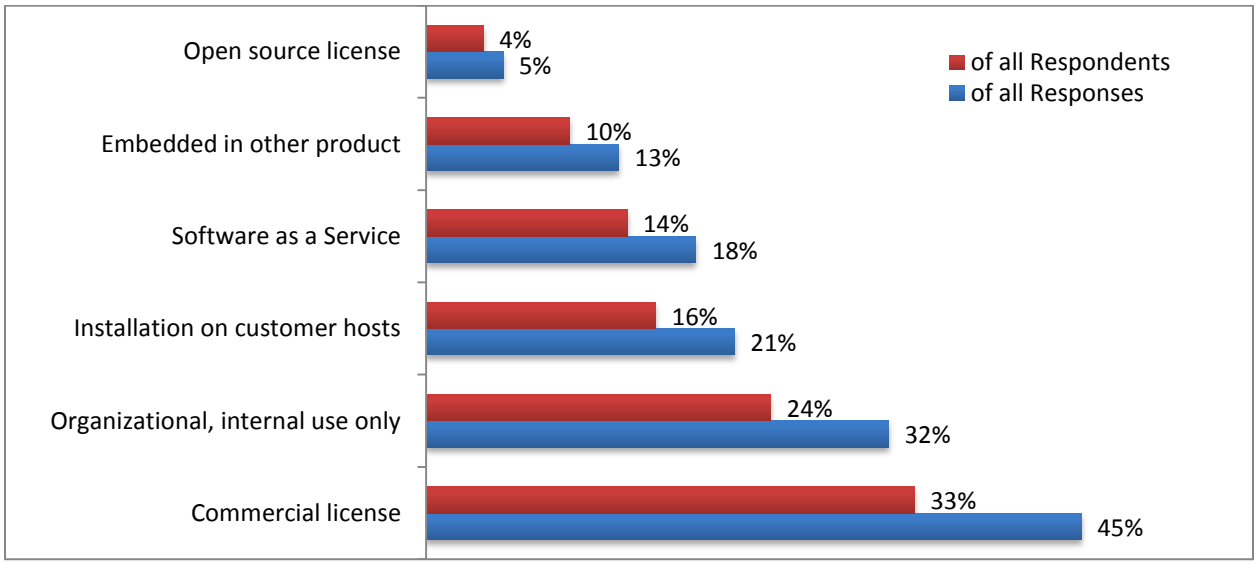


Figure 7 SUT Business Model

The terms and conditions under which software products are provided have broadened quite a bit in the last decade. As the economic imperatives are substantially different in each model, these responses characterize the extent to which a business model influences MBT usage. In this sample, the traditional business model dominates. However, 16% use two or more business models.

What kinds of computers are used in the SUT?

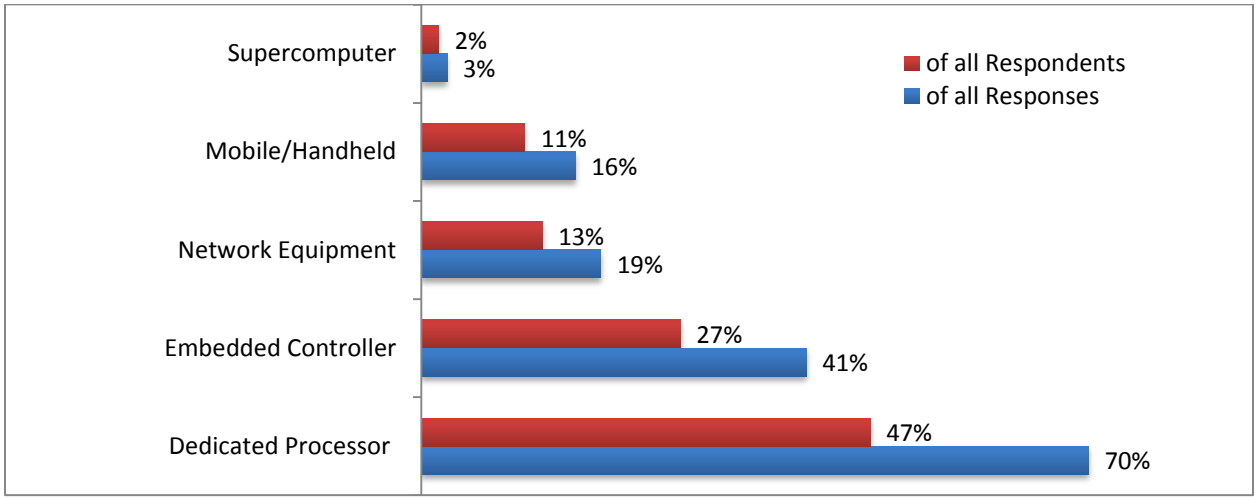


Figure 8 SUT Platforms

Figure 8 shows the extent to which MBT usage is associated with the general type of target platform. Nearly half develop for a standalone computer (desktop or server) and a little over a fourth develop for a platform embedded in another product. Twenty-nine percent report developing for multiple platforms.

What is the general architecture of the SUT?

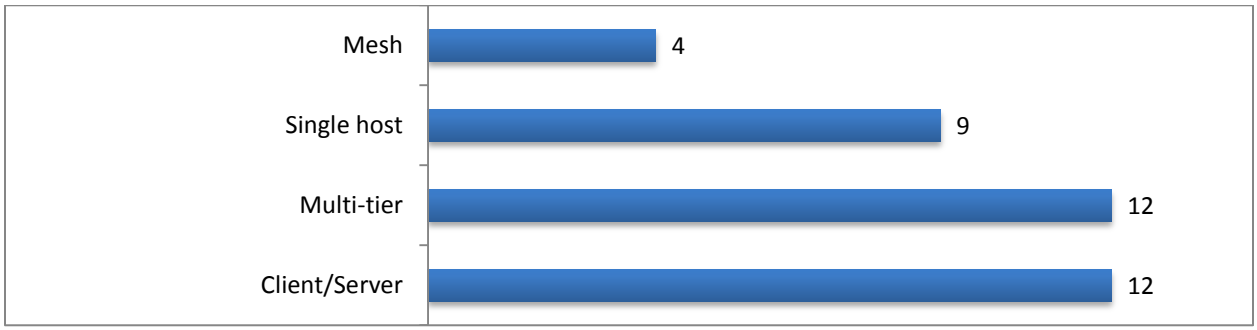


Figure 9 SUT Architecture

Figure 9 shows the extent to which MBT usage is associated with type of network architecture. In our sample, client/server and multi-tier applications dominate.

What OS(s) are used in the SUT?

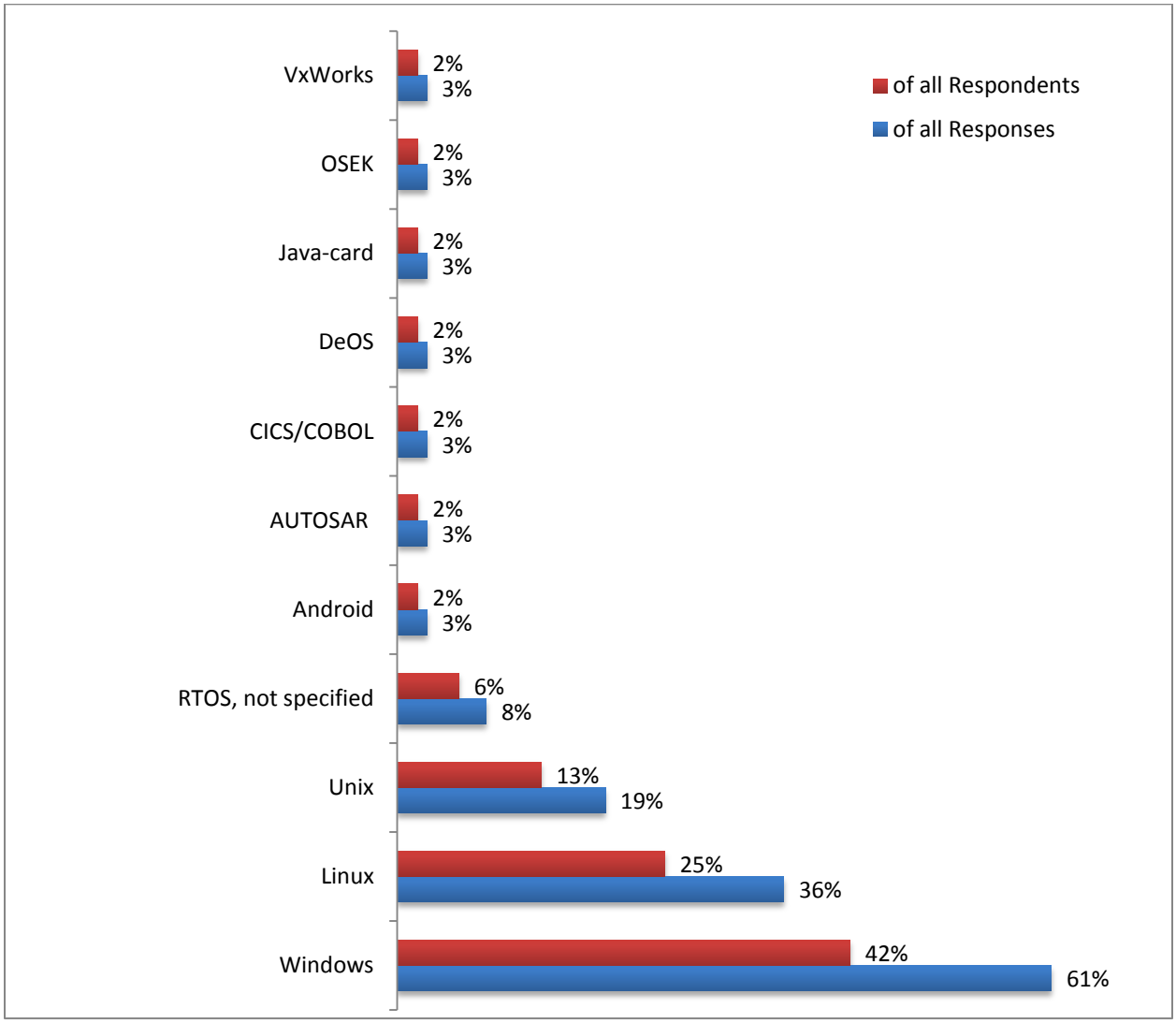


Figure 10 SUT Operating Systems

Microsoft Windows and *nix platforms are about equally represented in our sample. Twenty-six percent report developing for multiple operating systems. The embedded MBT users develop applications hosted on several kinds of RTOS.

In what programming language(s) is the SUT implemented?

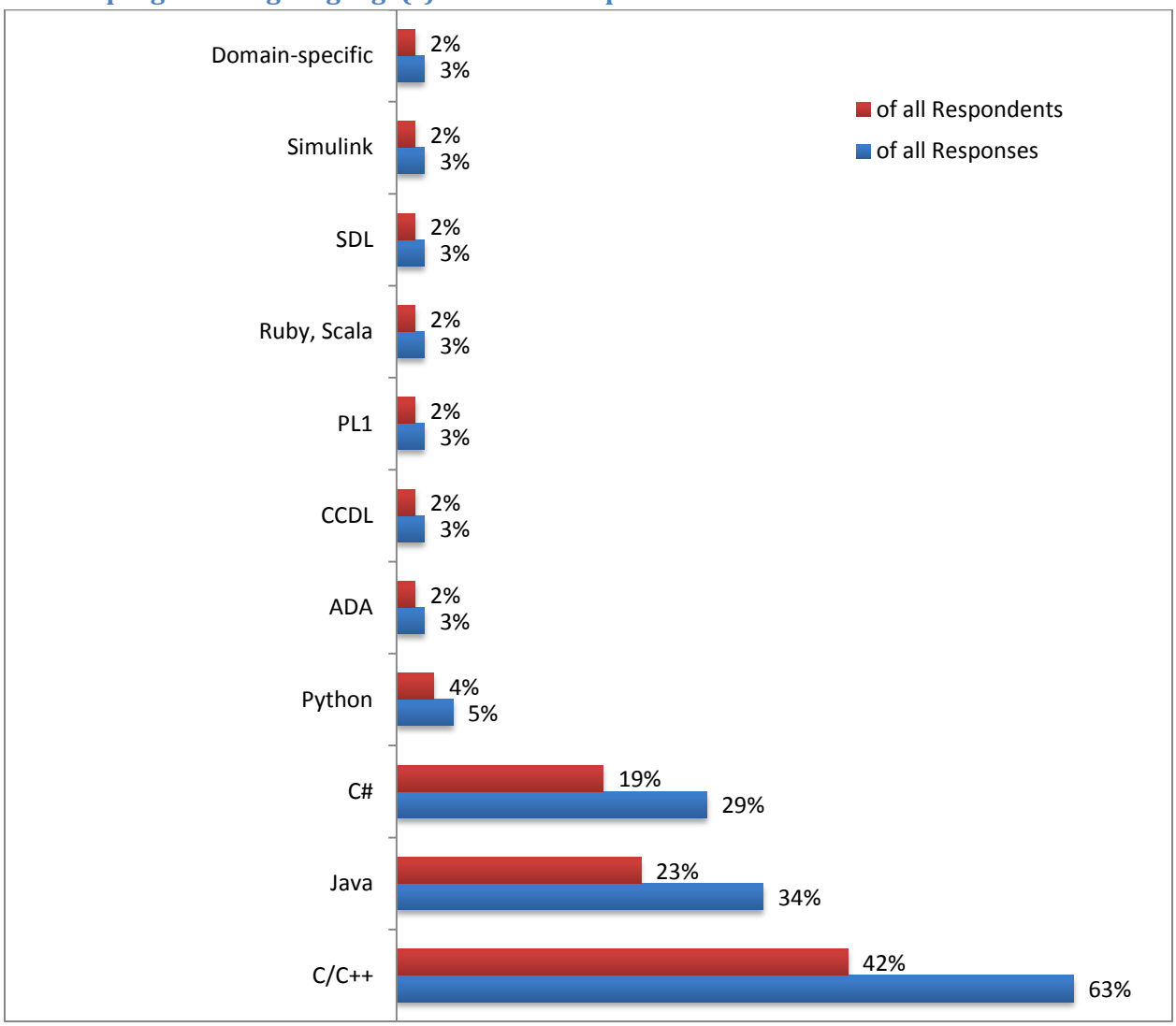


Figure 11 SUT Programming Languages

In our sample, programming languages C/C++ and Java dominate, with the remainder being several disparate types. No one reports using a functional language, which typically support AI applications. Twenty-nine percent report developing with multiple languages.

Impetus and Preparation

What factors led to your project using MBT?

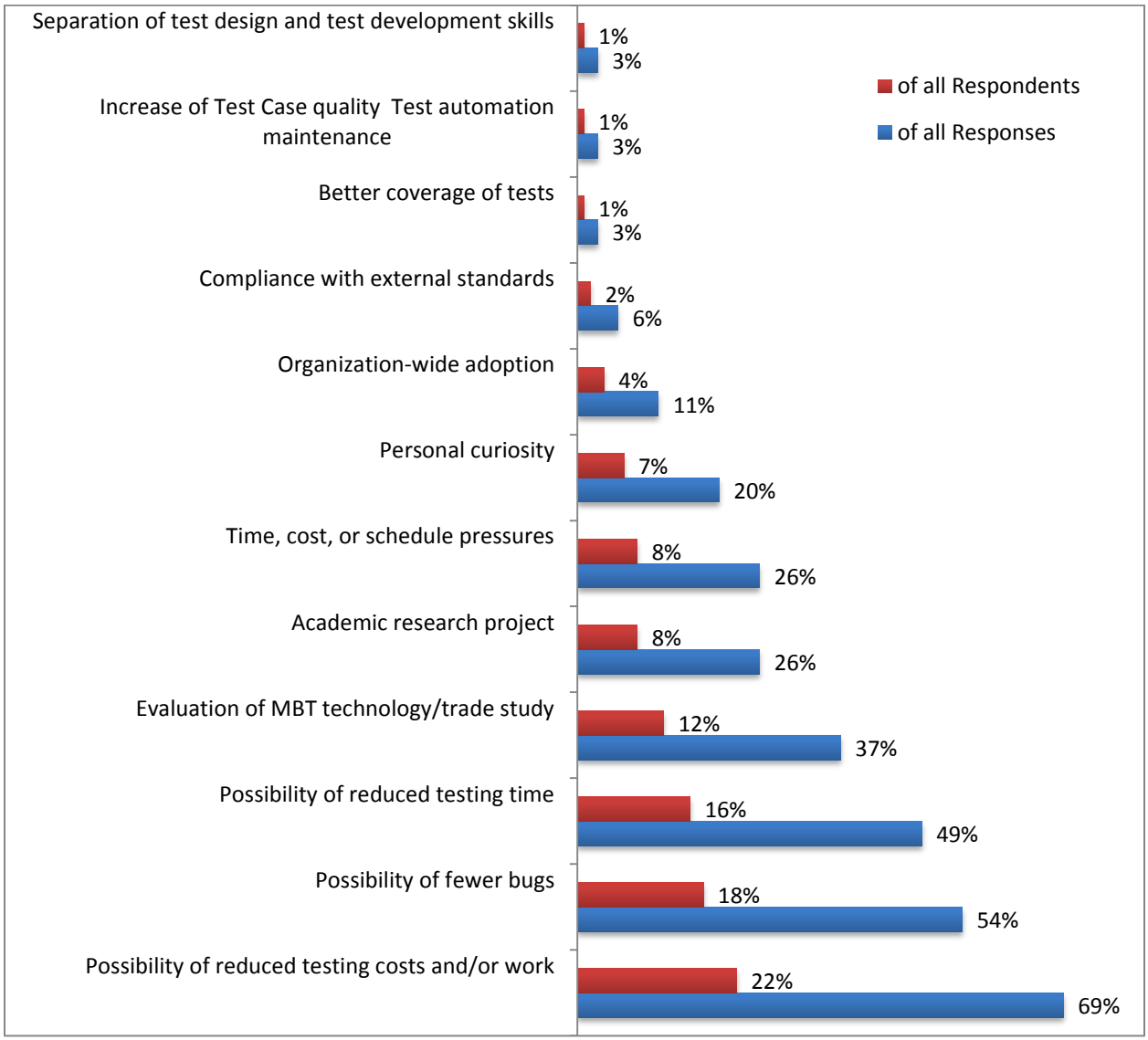


Figure 12 Impetus for using MBT

The leading motivations for using MBT are not surprising: desire for faster, better, and cheaper testing. About two-thirds of the respondents cited more than one motivating factor.

At what stage of MBT adoption is your organization?

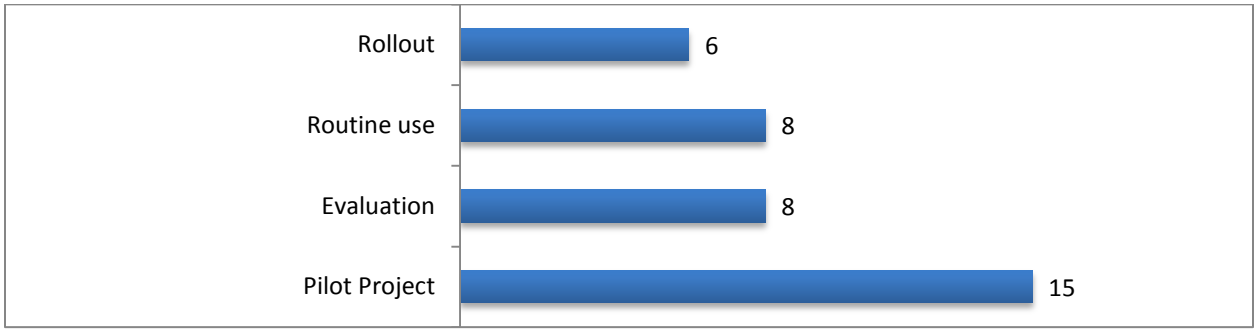


Figure 13 Stage of Adoption

Respondents are relatively new to using MBT: 62% are evaluating MBT or using it in a pilot project; 37% use MBT routinely or are rolling it out to others.

How did the MBT testers learn the skills necessary to model the SUT with the tool and generate test runs?

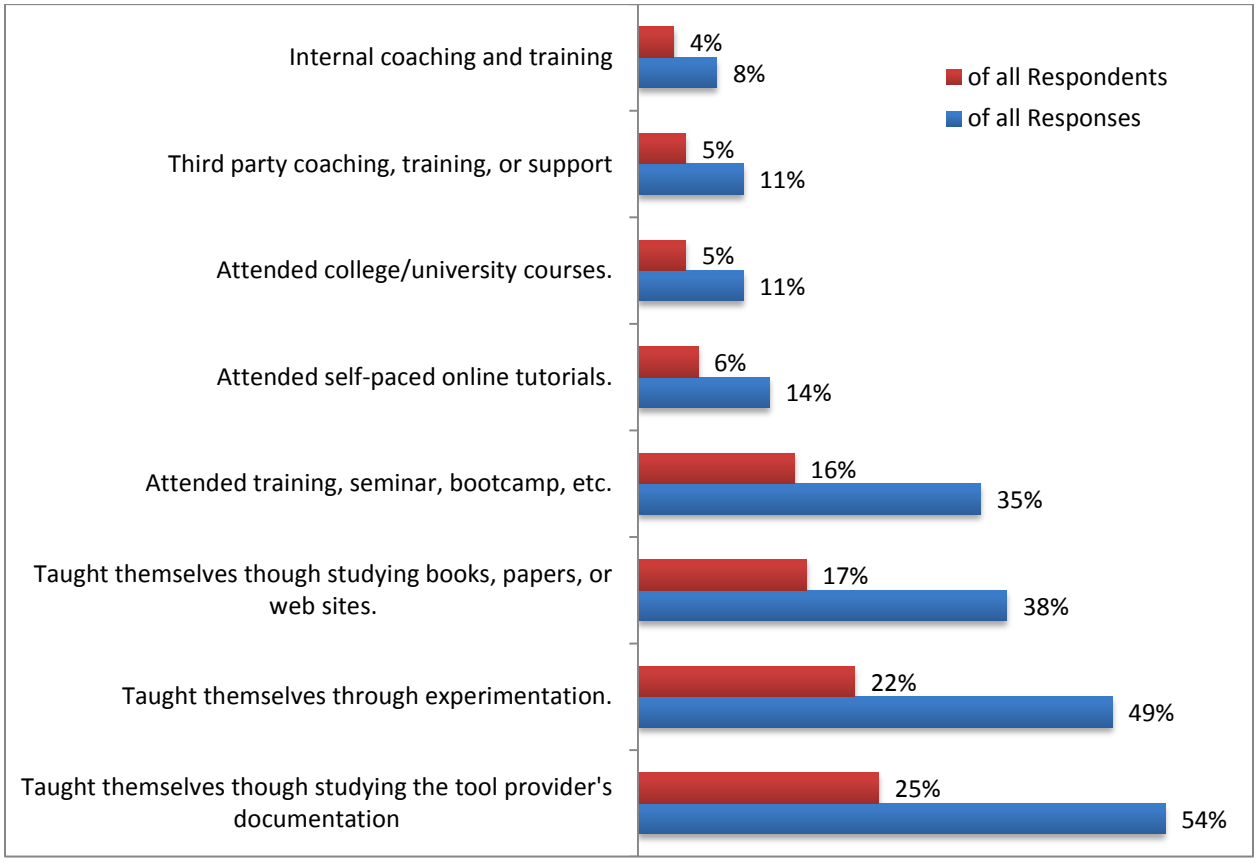


Figure 14 MBT Skill Development Approach

Although two-thirds of the respondents use two or more strategies, only 9% have supported their MBT projects with coaching and training.

On average, about how many hours of skill development were needed for a person to become a proficient user of the MBT tool?

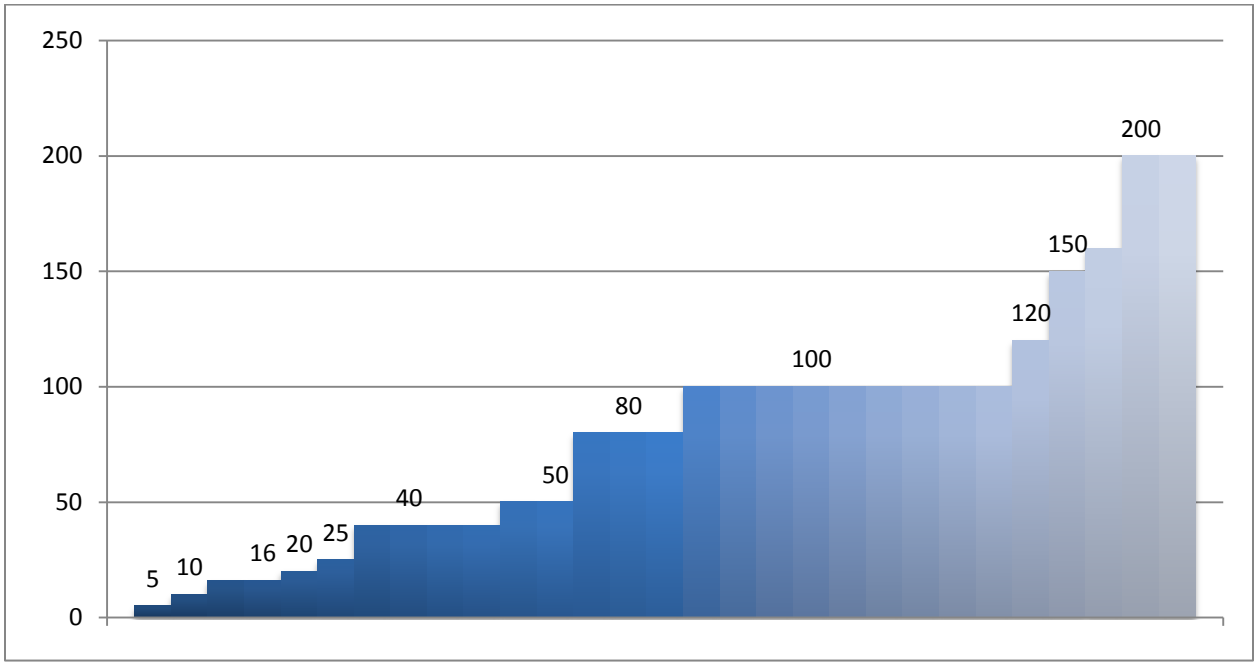


Figure 15 Average Training Time to Proficiency

The intellectual challenge of learning system modeling, the modeling language of a specific tool, as well as the tool's feature set can be a daunting task. Figure 15 shows how much effort respondents expended to become minimally proficient in the tool. The average and median time reported to attain minimal proficiency is 80 hours – about two working weeks.

MBT Usage and Configuration

What kind of MBT tools were used in the project?

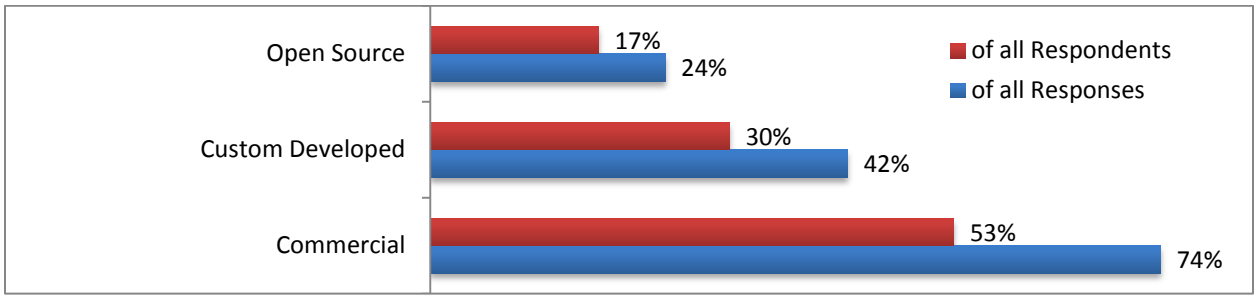


Figure 16 MBT Provider Types

Three in four respondents use a commercial MBT tool. Two in five use a proprietary solution. Open source tools are used by one in four. One in three reports using two or more provider types.

If you used a commercial tool, which one?

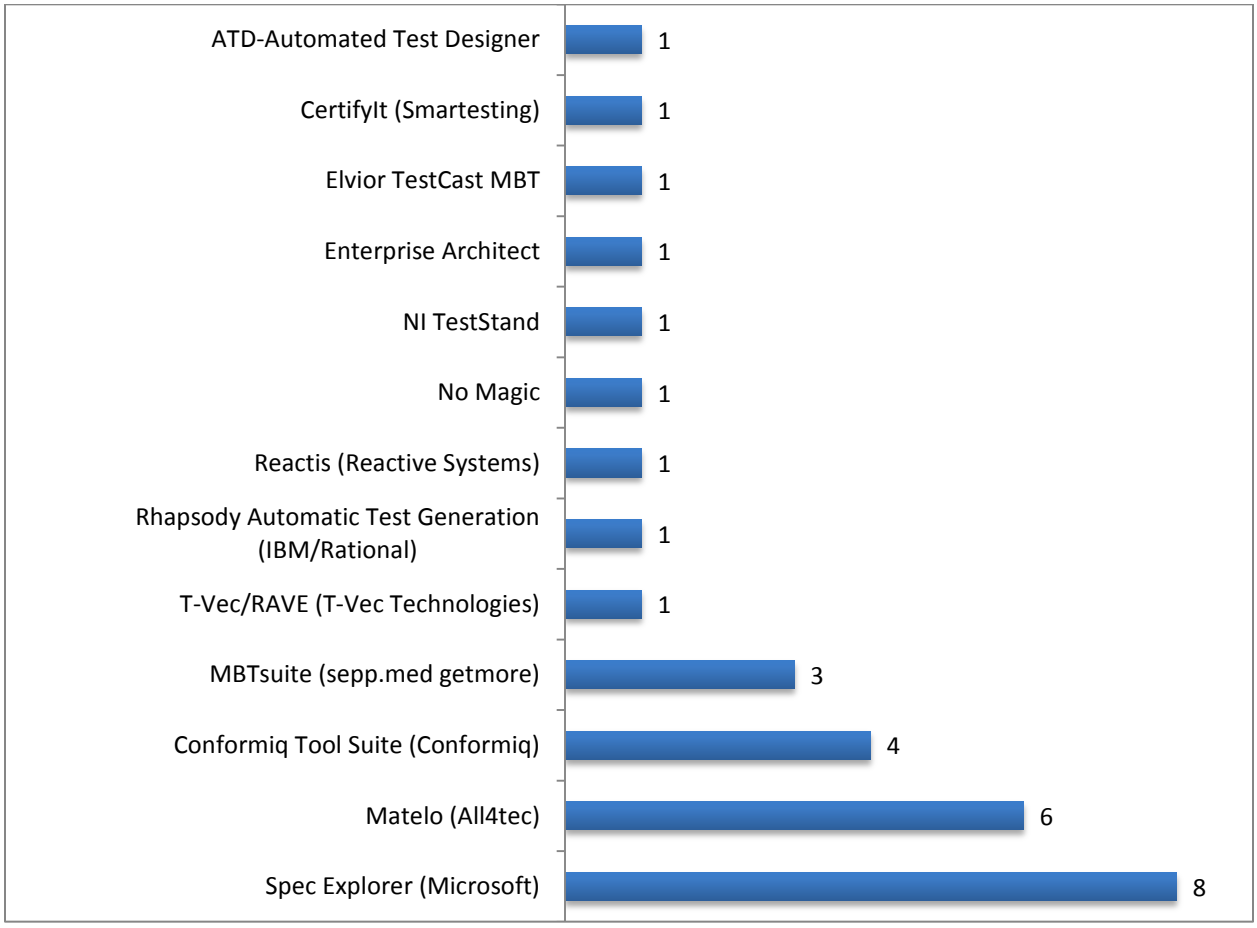


Figure 17 Commercial MBT Product Used

The respondents use products that include all established MBT tool providers, as well as entrants that are more recent. The proportions found in this survey are not indicative of provider market share.

If you used an open source MBT tool(s), please indicate which.

Nine respondents report using an open source MBT tool. Each used a different tool, listed in Figure 18. The proportions found in this survey are not indicative of provider market share.

IFx	http://www.irit.fr/ifx/
jBoss ; Eclipse (xtext; gPM; Topcased)	http://topcased.org/
JUMBL, PyUnit	http://sqrl.eecs.utk.edu/esp/jumbl.html
modeljunit, graphwalker	http://sourceforge.net/projects/modeljunit/
NModel	http://nmodel.codeplex.com/
OpenAmeos	http://www.openameos.org/about
OSMO	http://code.google.com/p/osmo/
Parteg, Pairwise (Microsoft)	http://parteg.sourceforge.net/
STSimulator	http://java.net/projects/stsimulator

Figure 18 Open Source MBT Systems Used

If you used an MBT tool(s) developed for use in your organization, please describe briefly.

Seven respondents reported developing proprietary MBT systems, as described in Figure 19.

Academic, research prototype
After evaluating several commercial and open-source tools, we developed our own tool based on EMF, M2M/M2T, and own test selection algorithms.
Check Case Definition Language (CCDL)
[For performance testing] we develop simple obviously correct slow algorithms, and check their results against the production code. Also, random testing of command line options based on a model of those options.
Supporting an automated test infrastructure for Hardware-in-the-loop (HIL) and Software-in-the-loop (SIL) testing
Use UML and business rules based models to derive test cases.
Modeling communication interfaces (APIs/protocols)

Figure 19 Proprietary MBT Systems Used

Does your MBT tool have useful integration (get/save data) with other development environment systems?

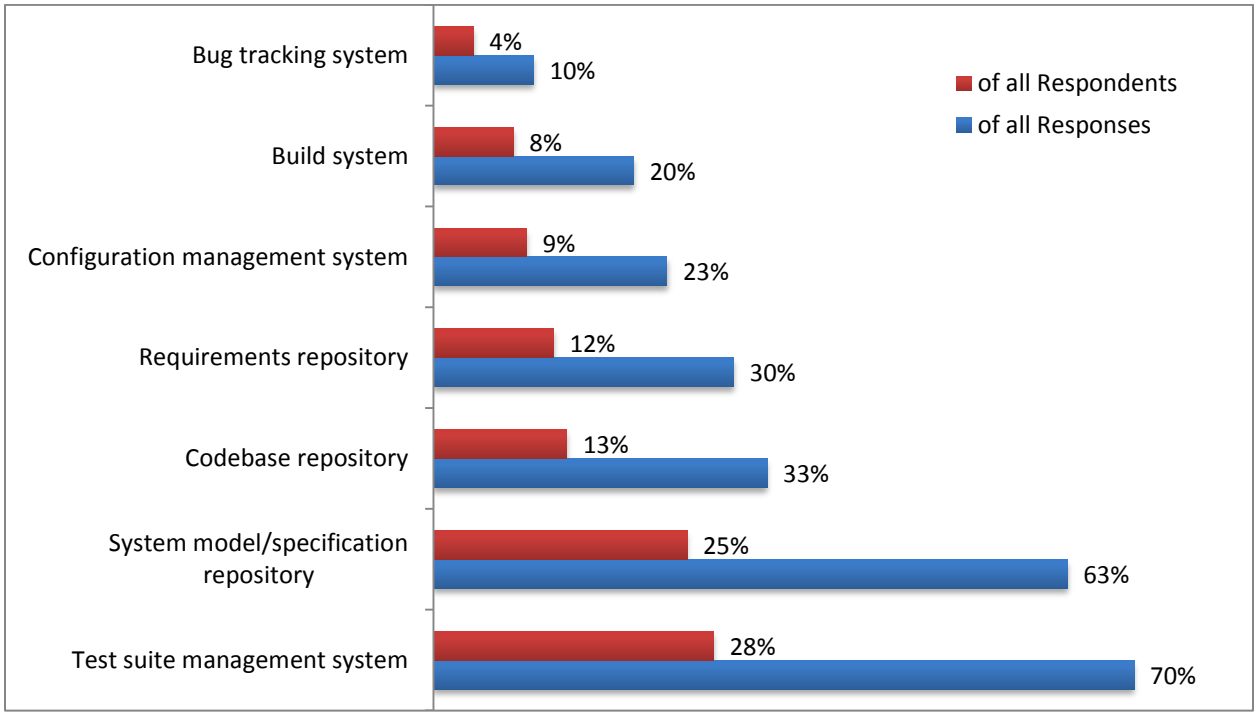


Figure 20 MBT/Environment Integration

About two in three report that MBT is integrated with their test asset management system as well as their requirements/design repository. Less than one in three report integration with other foundational development tools.

What is the proportion of work on model-based, traditional, and manual testing?

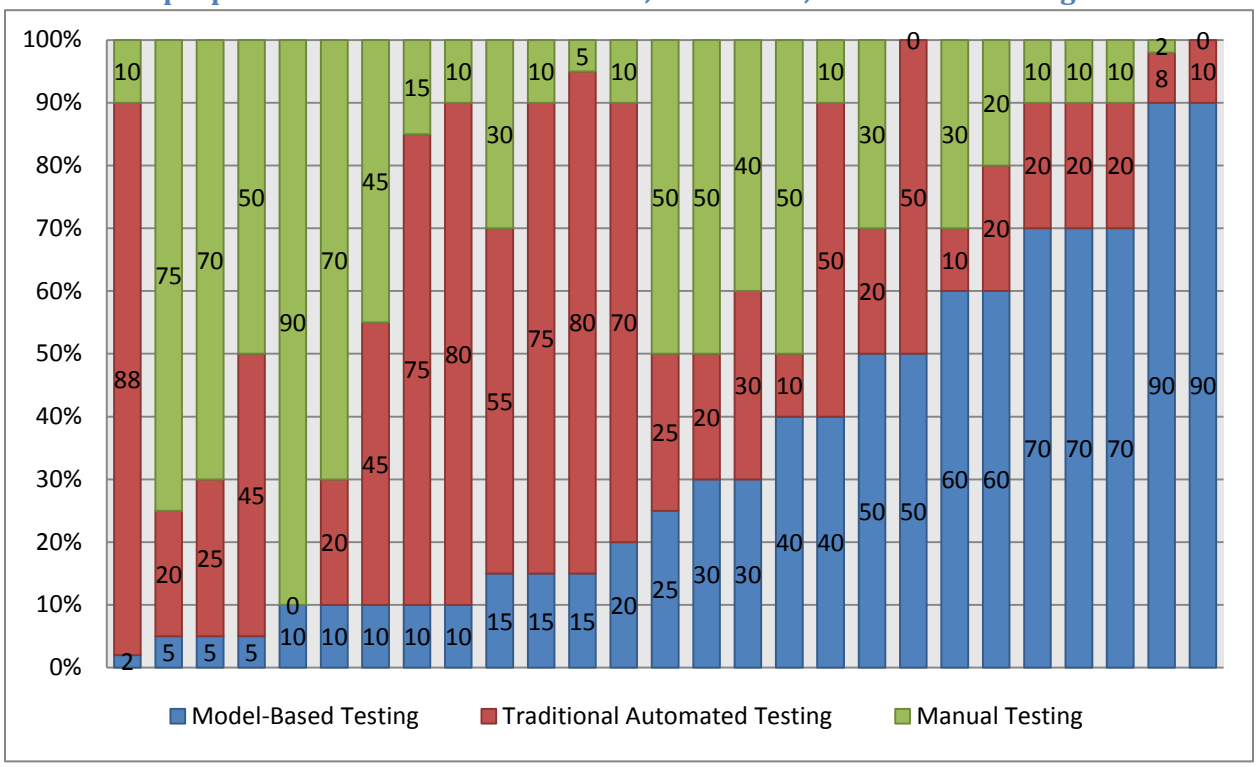


Figure 21 Proportion of Testing Modes

In manual testing, a person designs, selects, applies, and evaluates test cases. In traditional testing, a person designs and selects test cases that are then applied and evaluated with a programmable test harness. In MBT, a person develops a model using an MBT tool, which generates the test cases.

MBT is not the dominant testing mode for most of the respondents, as Figure 21 shows. Figure 22 shows that on average about 33% of all testing is model-based.

	MBT	Traditional	Manual
Average	33.6%	36.7%	29.7%
Median	25.0%	25.0%	20.0%

Figure 22 Average and Median Allocation by Testing Mode

To what extent is MBT testing integrated into your production build cycle?

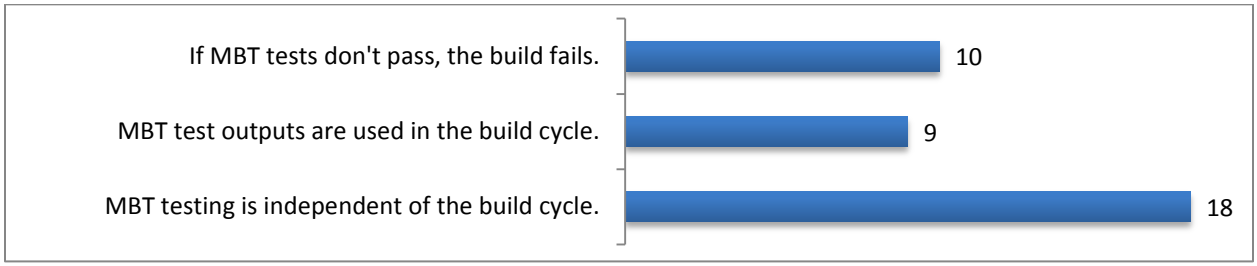


Figure 23 MBT Build Integration

It is a long-established best practice to run automated test suites to verify a new build and to discard the build when the test suites fail. One in four respondents use their MBT test suites in this way. About the same number use MBT test suites to support build verification. About half of the respondents do not use MBT suites for build verification.

Points of Control and Observation

The strategy used to submit test inputs (control) and then capture SUT response for evaluation (observation) is critical to making effective use of MBT-generated tests.

Figure 24 *MBT/Test Harness Integration*, shows how respondents couple MBT output with SUT interfaces. Fifty-seven percent couple the MBT tool with an abstract interface, which encapsulates the physical details of the actual SUT interface. This is generally accepted as a preferred approach. For a small proportion of respondents has a tester manually control and observe the SUT. Figure 25 and Figure 26 show which kinds of SUT interfaces are used to support test-binding strategy. GUIs are the second most frequently used interface type. This is a somewhat surprising (and encouraging), as controlling and observing GUIs is often fraught with difficulties.

What is the general approach for connecting MBT tool output, SUT control interfaces, and SUT observation interfaces?

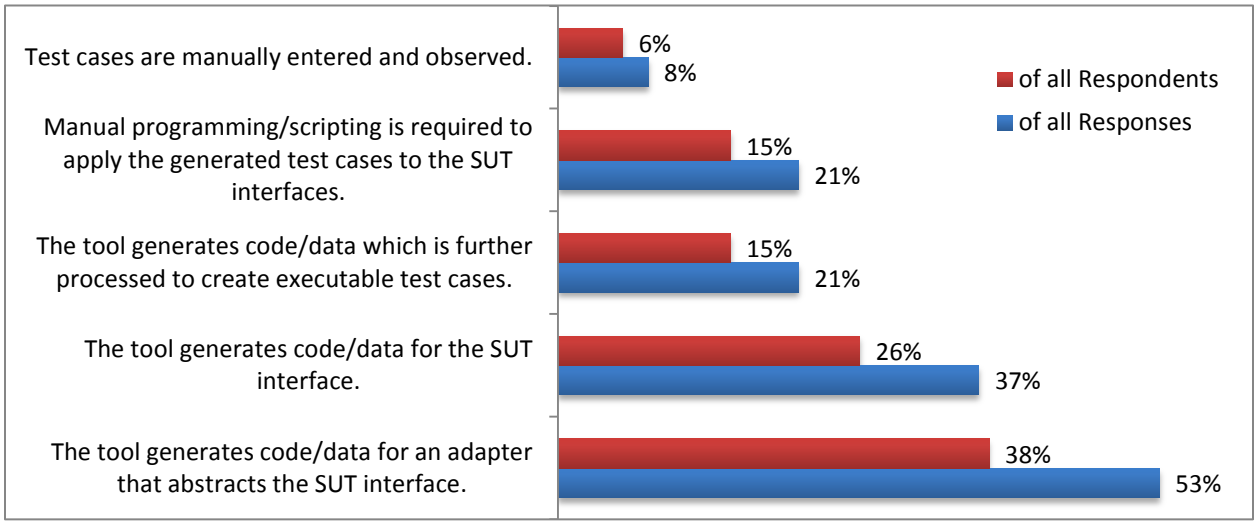


Figure 24 MBT/Test Harness Integration

What kind of SUT interfaces are controlled to setup and/or execute MBT test cases?

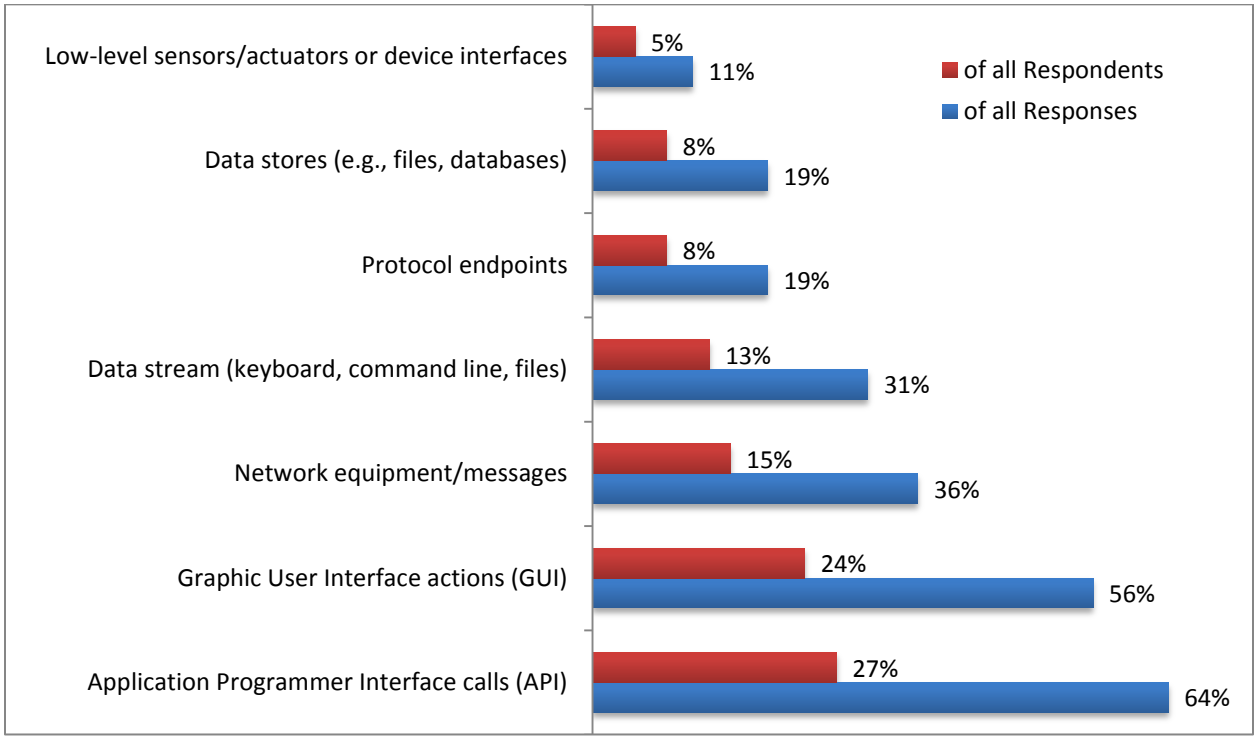


Figure 25 MBT Points of Control

What kind of SUT interfaces are observed to provide test responses for automated MBT evaluation?

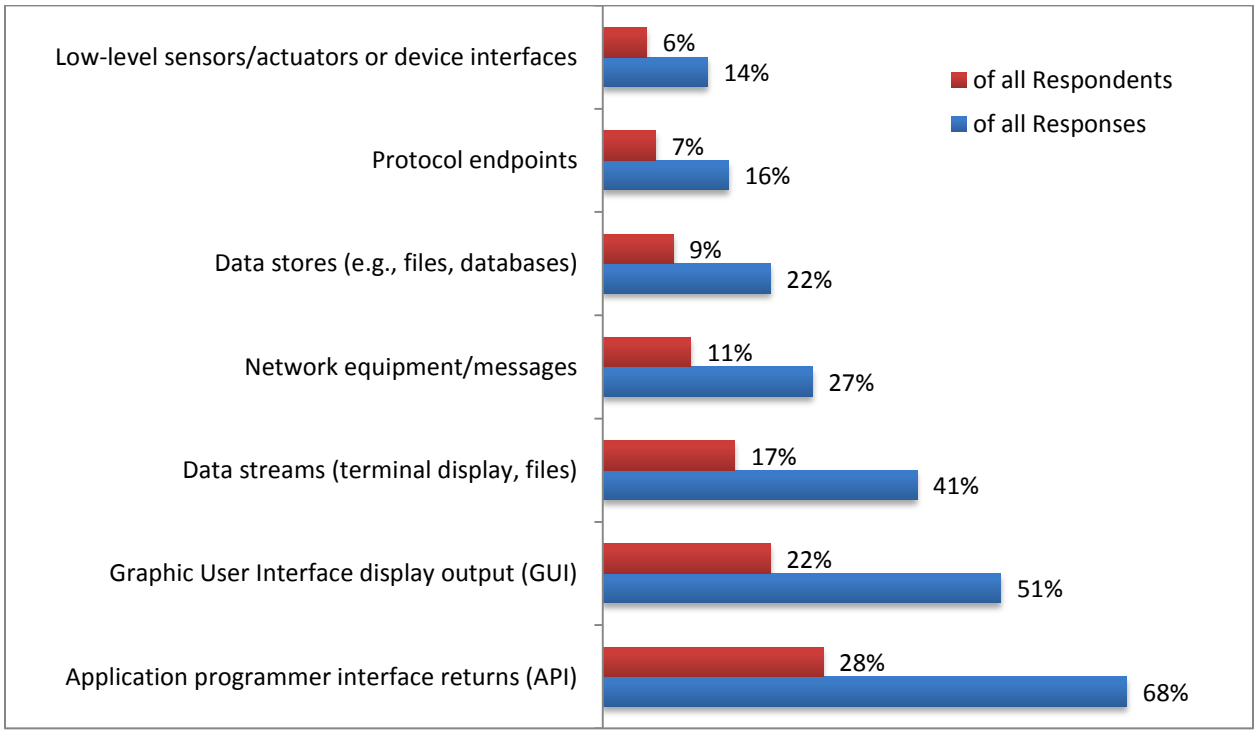


Figure 26 MBT Points of Observation

Testing Practices

Which aspects of the SUT do you test with MBT?

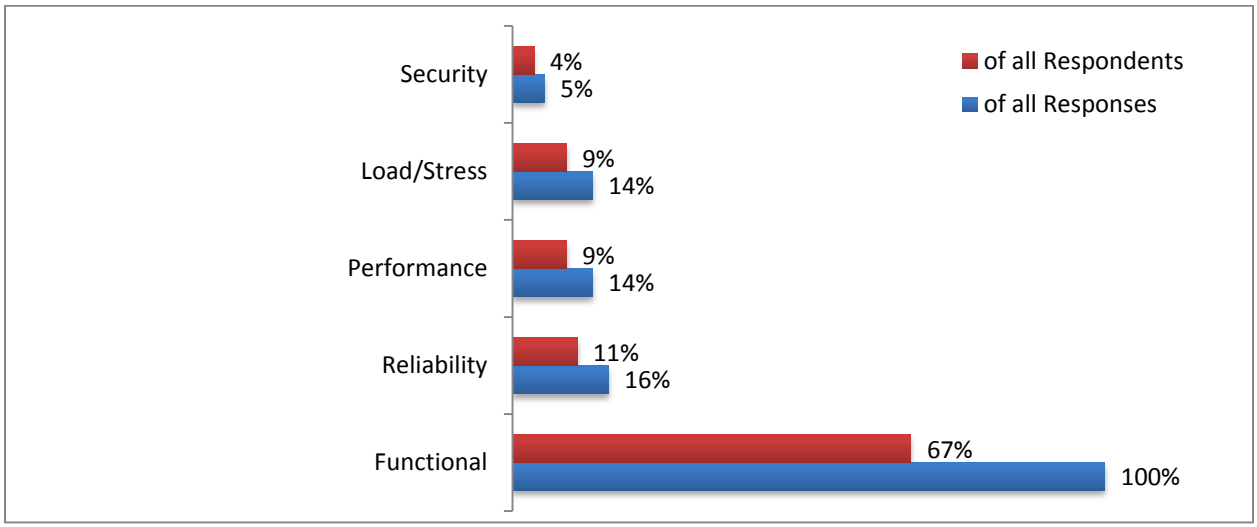


Figure 27 Aspects Tested with MBT

All respondents use MBT to evaluate functionality. About one in ten use MBT to evaluate other "ilities."

At what scope do you use MBT?

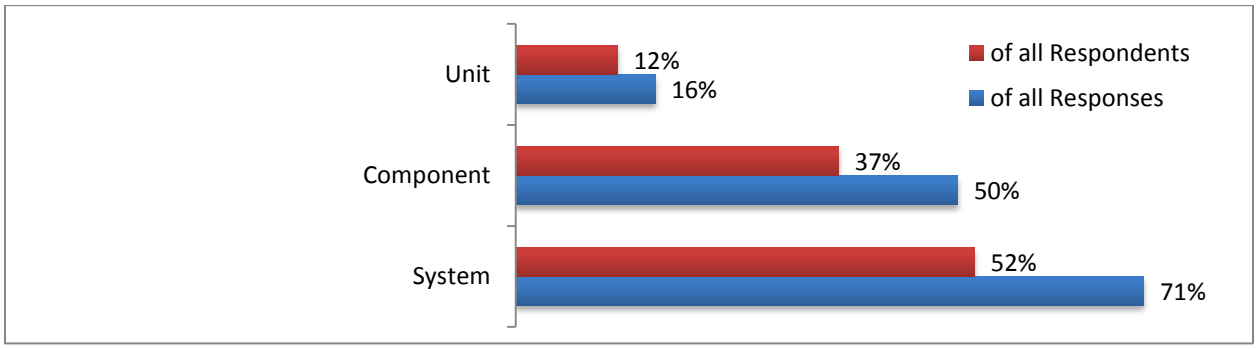


Figure 28 Scope of Testing with MBT

MBT is used at all scopes of testing: primarily at system scope, at component scope for half of the respondents, and at unit scope for 16%.

What kind of coverage is measured for your MBT test runs?

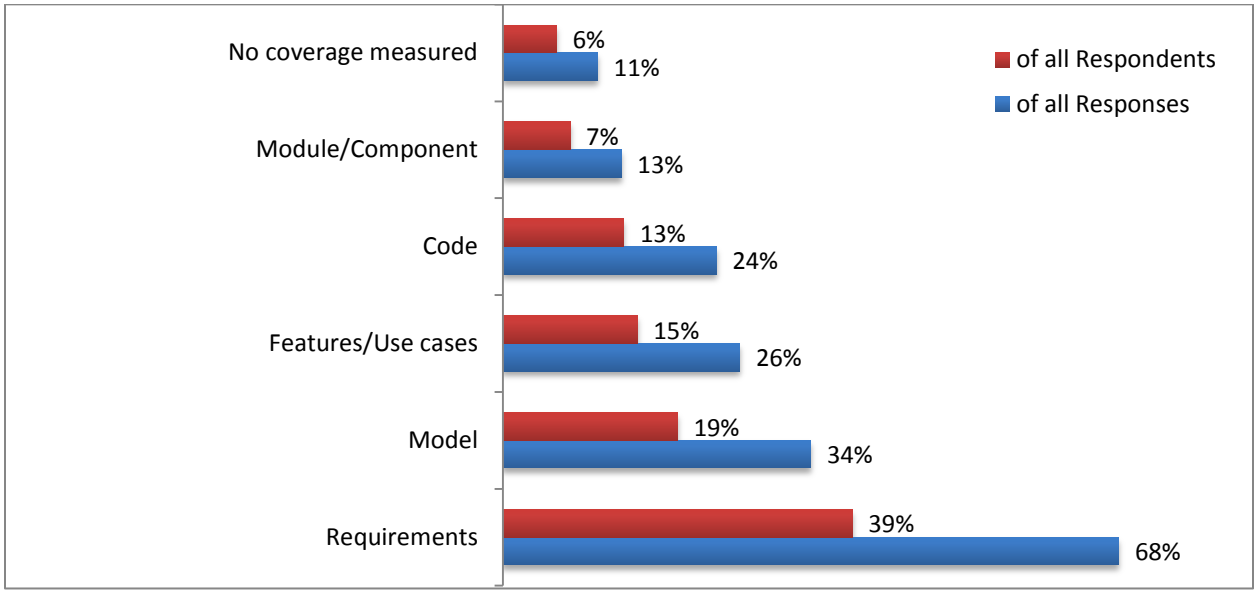


Figure 29 Test Coverage Measured

Two in three users measure requirements coverage, with about half as many measuring model coverage. About one in four measures feature or code coverage. One in ten does not measure any kind of coverage resulting from MBT test runs.

Time, Cost, Quality, and Effectiveness

Overall, respondents view MBT as an effective technology (Figure 35 Overall *Effectiveness of MBT*.) Problems that hampered adoption of first generation tools are now viewed as manageable (Figure 30.) Most report improvements in quality as well as the time and cost of testing (Figure 31, Figure 32, Figure 33.) Co-workers have a generally favorable view of MBT (Figure 34.)

What has your experience been with some common MBT problems?

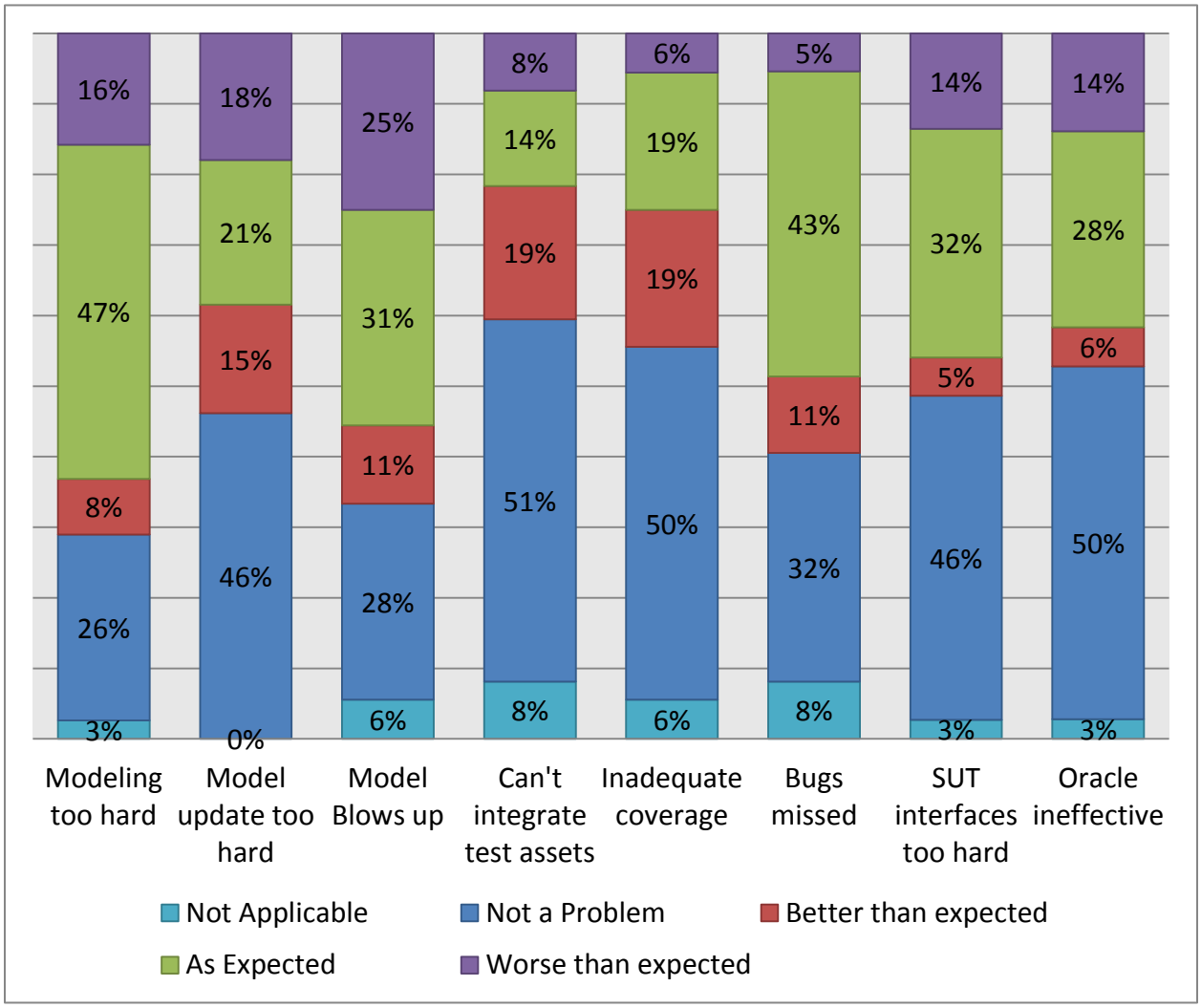


Figure 30 Impact of Common Problems

A 2006 study found recurring problems that often lead to abandoning MBT.[4] Figure 30 shows the extent to which respondents have encountered these problems and others commonly recognized in the MBT community. In varying proportions, other users report they have encountered these problems ("as expected".) About one in six reports they are worse than expected.

[4] Alan Hartman. "Ten Years of Model Based Testing: A Sober Evaluation," *MBT 2006,* IBM Israel – Haifa Research Laboratory.

What is your best estimate of the percent increase or decrease in escaped bugs after testing with MBT?

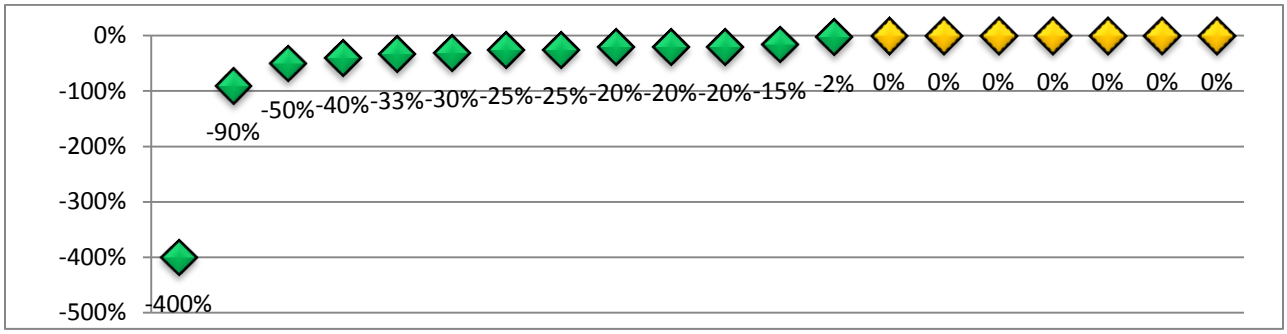


Figure 31 Effect of MBT on Escaped Bugs

Respondents report that on average bug escapes decreased by 59%, with the median improvement at 25%. No one reported an increase; seven respondents reported no change.

What is your best estimate of the percent increase or decrease in testing costs with MBT?

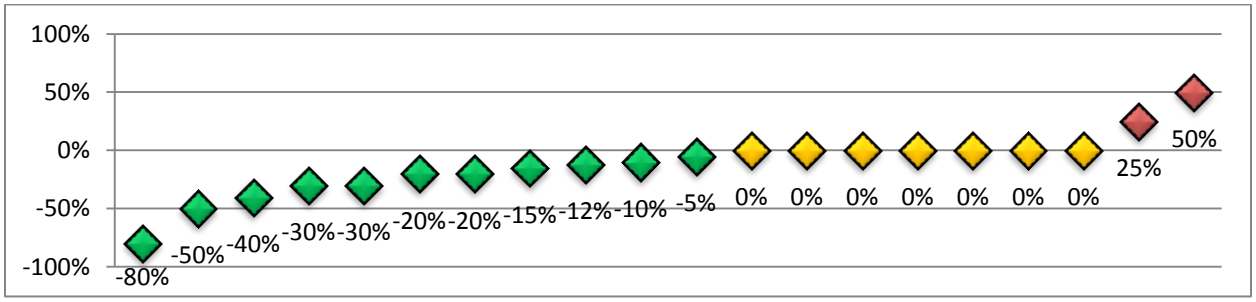


Figure 32 Effect of MBT on Testing Costs

Respondents report that on average testing costs decreased by 16%, with the median improvement at 15%. Seven saw no difference; two reported an increase in costs (25% and 50%).

What is your best estimate of the percent increase or decrease in testing duration with MBT?

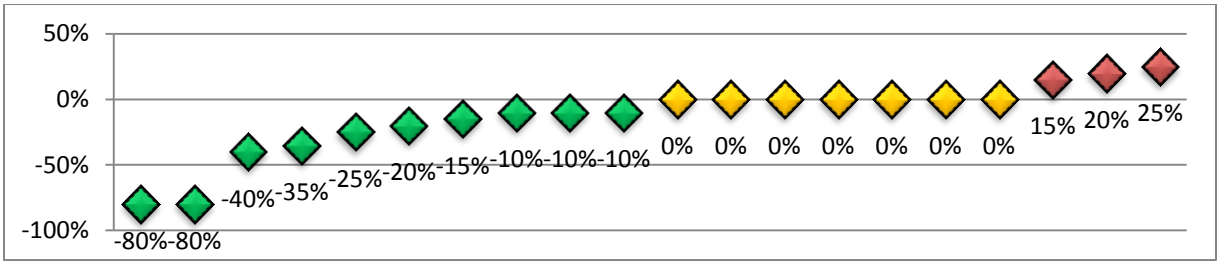


Figure 33 Effect of MBT on Testing Duration

Respondents report that on average testing duration decreased by 25%, with the median improvement at 18%. Seven saw no difference; three reported an increase in time around 20%.

In your opinion, how do others view the overall effectiveness of MBT?

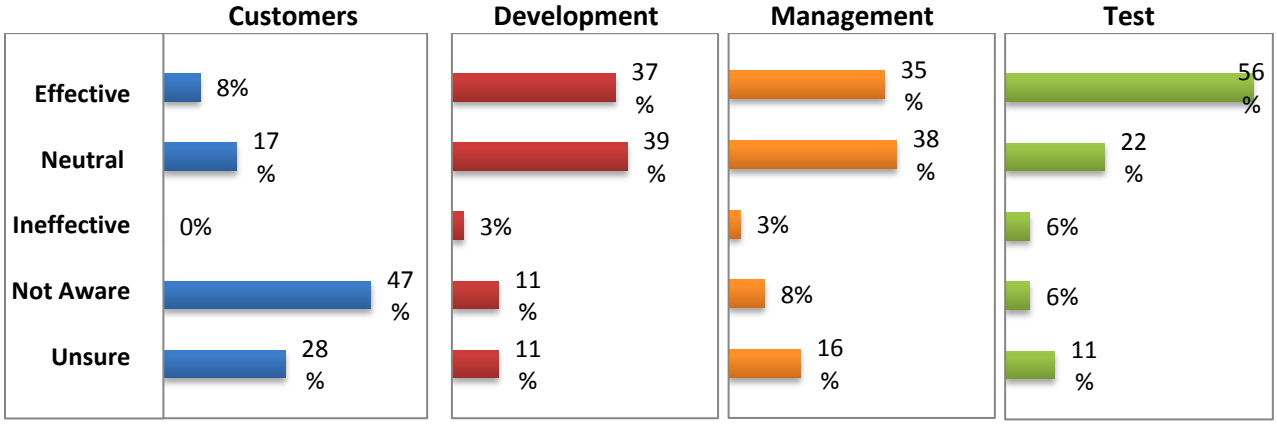


Figure 34 Views of others on MBT Effectiveness

The perceptions of stakeholders in the software process are often an important factor in acceptance of software technologies. MBT users report that software developers and managers in their organization are about equally likely to have a favorable or neutral view of MBT, and that other testers are twice as likely to have a favorable view. Only a few respondents report that other stakeholders views MBT as ineffective. With the exception of customers/users, most respondents indicate that other stakeholders are aware of MBT.

The responses in Figure 34 suggest that to the extent that other stakeholders are aware of MBT, they view it as either effective or neutral. Respondents were instructed to tick the "unsure" category if *they* were unsure of others perceptions (it does not indicate that, for example, *customers* are unsure about MBT.)

Overall, how effective do you think MBT has been?

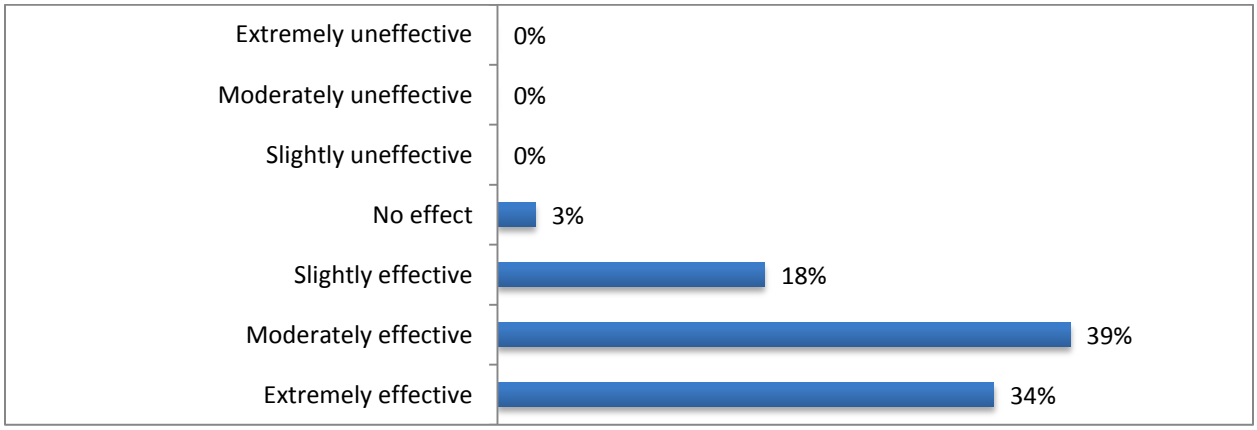


Figure 35 Overall Effectiveness of MBT

Three out of four respondents see MBT as either moderately or extremely effective. No respondent rated MBT as ineffective, in any degree.

About one in five views MBT as “slightly effective,” with only one reporting “no effect.” Only one of these respondents also provided an answer about effects on time cost and quality (Figure 31, Figure 32, Figure 33.) It is not clear these respondents did not quantify their less favorable assessment. However, it is clear that the more-favorable group has quantitative clarity about the effects of MBT.

Future Use

How likely are you to continue to use MBT?

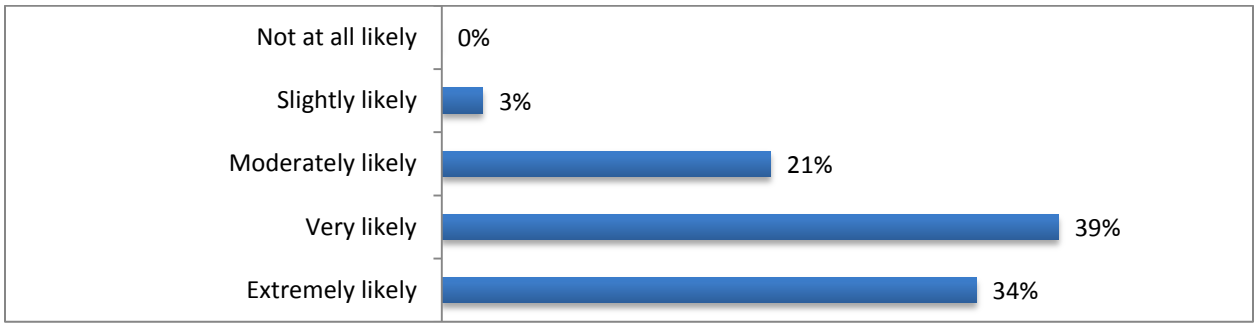


Figure 36 Planned Future Use

Describe the biggest limitation of MBT.

Respondents provided the following observations about MBT limitations.

Value	Price of commercial tools.
	I don't see any limitation: If you can't model it - you can't test it!
Modeling	Too difficult to determine right level of detail. If too detailed or not detailed enough, it's too hard to generate test cases, which requires rework to models.
	Getting valid models.
	Finding the right model notation/paradigm to capture the desired aspect of SUT behaviour.
Process	Low coupling with process, lack of skills, and need for changed process.
	Lack of understanding and support in industry, leading to difficulty integrating with existing test methods.
	Management is not fully aware of growing complexity of SUT and of demanding technologies needed to test with it.
	Lack of commitment to product reliability.
	Awareness and integration.
Requirements	Test results analysis when doing test focusing on requirements.
	Lack of complete requirements limits model validity.
	Lack of good requirements or models.
Technology	Model debugging.
	Model management and adequacy of modeling aspects; expressiveness of models.
	Tool integration capabilities.
Training	Learning a complex tool-specific modeling paradigm.
	Too formal and not understandable for new users.
	Training effort and change to current roles.
	Needed experience in following technologies: constraint solver, constraint programming, modeling of timed systems, exploration, compositional architecture for test models.
	Good modeling skills are a constraint; tools are not useful without that.

Figure 37 Statements about MBT Limitations

Describe the biggest advantage of MBT.

Respondents provided the following observations about MBT advantages.

Effectiveness	Communication, test automation, systematic test activation.
	Automated test generation.
	Model transformation capabilities.
	Abstraction as opposed to hardcoded scenarios.
	Test automation.
	Extremely thorough testing and easy to modify test sequences.
Quality	Real product coverage.
	Understanding software more thoroughly and getting better test cases that truly test the structure of the software.
	Better coverage, time saving and cost efficient especially for multiple configurations.
	Every test run has brand new test case. The coverage and tested combinations are rising all the time.
	Complements implementation to match hard requirements.
	Design bugs detected while developing the model.
	Issues are detected earlier (test involved earlier).
	Better coverage of test cases.
Time	Agility to changes and improved understanding and sharing of test activities within the project team.
	Agile way to perform test design.
	Maintainability of generated test cases, higher test coverage.
	Better coverage, time saving and cost efficient especially for multiple configurations.
Value	Give testing a systematic foundation, MBT is applicable to many domains. Reuse. Usage of only a few MBT-tools, less in-house tool development.
	Maximizing testing thoroughness within resource constraints.
	Higher quality, lower costs.
	Reducing testing effort and increasing test automation.
	Better coverage, time saving and cost efficient especially for multiple configurations.

Figure 38 Statements of MBT Advantages

Additional comments about this survey or MBT.

Respondents provided the following general observations about MBT.

MBT needs more standards to help determine right level of detail. Also, need more MBT tools to help auto-generate test cases.
Need more features and process related coupling in MBT tools.
MBT is only as good as the models are. Unfortunately, there is not yet a fundamental support on model management or guidelines at which level modeling should happen (balancing design and implementation effort).

Figure 39 General Statements about MBT

Selected Publications

The following sources provide overviews and details of MBT theory, practice, and technologies.

Mark Utting and Bruno Legeard. *Practical Model-Based Testing: A Tools Approach*. Morgan Kaufmann, 2006.

Jonathan Jacky, Margus Veanes, Colin Campbell, and Wolfram Schulte. *Model-Based Software Testing and Analysis with C#.* Cambridge University Press, 2007.

Justyna Zander, Ina Schieferdecker, and Pieter J. Mosterman. *Model-Based Testing for Embedded Systems*. CRC Press, 2011

Neto et al, "Improving Evidence about Software Technologies: A Look at Model-Based Testing," *IEEE Software*, vol. 25, no. 3, pp. 10-13, May/June, 2008.

About System Verification Associates

System Verification Associates, LLC is based in Chicago and lead by Robert V. Binder. We provide the following services:

- Software process assessment
- Design/build advanced automated testing environments
- Design/build model-based testing solutions
- Reliability analysis
- Expert reviewing
- Methodology development and support
- Regulatory compliance review and remediation
- Quality management for outsourced development

Our clients develop and market some of the most complex software and communication systems ever built. They face substantial challenges in reliability, product complexity, outsource management, time to market, and regulatory compliance. Our services have been an integral part of their solutions to these problems.

For more information, please visit http://sysvas.com or email Robert V. Binder at rvbinder@gmail.com.