A Preliminary Survey on Software Testing Practices in Australia[1]

S.P. Ng[*], T. Murnane[†], K. Reed[†], D. Grant[*], T.Y. Chen[*]

[†]*School of Engineering & Mathematical Science*
La Trobe University
Kingsbury Drive, Bundoora 3086
Australia
Email: {t.murnane, k.reed}@latrobe.edu.au
Phone: (+613) 9479 1377
Fax: (+613) 9479 3060

[*]*School of Information Technology*
Swinburne University of Technology
John Street, Hawthorn 3122
Australia
Email: {sng, dgrant, tchen}@it.swin.edu.au
Phone: (+613) 9214 5505
Fax: (+613) 9819 0823

Abstract

This paper presents the findings of, to the best of our knowledge, the first survey on software testing practices carried out in Australian ICT industry. A total of 65 organizations from various major capital cities in Australia participated in the survey, which was conducted between 2002 and 2003.

The survey focused on five major aspects of software testing, namely testing methodologies and techniques, automated testing tools, software testing metrics, testing standards, and software testing training and education. Based on the survey results, current practices in software testing are reported, as well as some observations and recommendations for the future of software testing in Australia for industry and academia.

Keywords: *Software engineering, software testing, survey*

1. Introduction

Swinburne University of Technology, in conjunction with La Trobe University and sponsored by the Australian Computer Society conducted a survey on software testing in Australia between 2002 and 2003. Similar surveys are being run in several other Southeast Asian countries. Software development organizations from different industry sectors (government, pubic and private), domestic and foreign owned, in-house groups and software companies across various industries were invited to participate in the survey.

There were a number of reasons for conducting this survey:-

Firstly, anecdotal evidence from software developers suggests that testing is becoming an increasing percentage of the development budget.

Secondly, the authors' view is that software quality will become an increasingly important factor in software marketing. As this evolves, testing strategies will (in our view) become progressively more important. A carefully constructed survey has the potential of identifying the best practices, which can then be disseminated.

Thirdly, the survey may provide indications of future research directions.

Fourthly, the comparison with parallel surveys in the region will assist all national industries to both improve software quality and identify optimum testing strategies.

Finally, the results will provide guidance for those training software developers and software engineers.

The observations reported in this paper were based on 65 respondents successfully completing the questionnaire. Interestingly, the results from analyzing these 65 responses follow almost the same trends obtained from an earlier analysis performed three months ago using the first 41 responses. Despite the relatively small sample population in the survey, the consistency of the data obtained heightened our confidence to report the observations in this paper.

The remainder of this paper is structured as follows. Section 2 explains the methods that were used to plan and conduct the survey, including the method of selecting a research sample, the variables that the survey aimed to measure, the approaches used to invite subjects, and the methods of collecting data from respondents. Section 3 reports and discusses the results of the survey, including organization information of the respondents. Section 4 analyses and summarizes the survey findings, and discusses the implications of the survey on the software testing industry, as well as its implications on training and

[1] This Survey was funded by the Australian Computer Society under its research program.

education of software testing personnel, both in the workplace and at universities. Section 5 concludes the paper and suggests future work.

2. Survey Methodology

2.1 Survey Objectives

Two of the five objectives listed in the introduction were used as the design objectives for this survey, since the others are considered as outcomes that flow from these. The primary objective was to determine the types of testing techniques, tools, metrics and standards that organizations in Australia use when carrying out software testing activities (this of course embraces several of the listed objectives). The purpose of this was to provide a concise picture of the current industry best practices.

The second objective was to determine whether existing training courses in software testing taught in the workplace or in similar study at tertiary institutes adequately cover the types of testing methodologies and skills that industry requires. If these requirements were not met, the industry may benefit from the survey recommendations to address any deficiency observed and ultimately improve the existing training opportunities available to practitioners as well as novice testers.

Based on these two objectives, a number of hypotheses were employed to design the questionnaire and shape the direction of the survey.

2.2 Survey Description

The survey targeted senior employees involved with testing in software development organizations. Requests were addressed to software testing or project mangers as the personnel most likely to understand their testing environments and experiences within their organization.

Five major areas of software testing related activities were investigated by the survey. In addition, an introductory section was also included to assess the organization size and structure, and where relevant, history of the organization and its overall procedures with respect to software development and testing. Using the conjectures in our hypotheses as means of constructing specific questions, the questionnaire was arranged into the following six sections. The information sought can be summarized as follows.

Section A - Organization Information

This section captured the type and size of the organization, including specifics such as the current number of general employees and IT professionals, the number of applications developed and tested over the past three years, the allocated and actual budget for testing among the other various software development activities, as well as questions relating to whether the organization wrote specifications and whether changes to specifications were controlled and tracked.

Section B - Software Testing Methodologies and Techniques

The extent to which software testing methodologies and general testing techniques are used in the industry and the current practices of those organizations adopting structured methodologies and techniques in software testing were investigated in this section.

Section C - Automated Software Testing Tools

Questions relating to the extent to which automated testing tools are used in industry, including commercial and in-house developed tools, were placed in this section revealed. The level of satisfaction with such tools was assessed by querying the respondents' belief that the quality of developed software was being improved by the use of such tools.

Section D - Software Testing Metrics

This section explored the extent to which software testing metrics are used by industry, and if and how those metrics are improving the quality of software under development.

Section E - Software Testing Standards

The usage of standards for software testing in industry, including published standards such as ISO, CMM and their quality accreditation, as well as in-house developed standards was assessed in this section. Questions were posed to determine whether the use of standards was considered to improve the software development processes of the organization.

Section F - Software Testing Training and Education

This section determined the extent to which organizations provide training in software testing for their employees. Also examined was the organization's view on the factors that attract software testing staff to attend training courses as well as the benefits for testing staff that accrue. The usage of various sources of training courses (such as universities or TAFE colleges, external commercial training courses, in-house training and self-study) were also queried.

2.3 Survey Method

A questionnaire comprised of both closed and open-type questions was used. Survey interviews were conducted face to face, over the telephone, via facsimile or email attachment. To allow for more flexible arrangements, some respondents were invited to complete the online questionnaire at our survey web site[2]. In all cases, printed or verbal explanatory notes were provided to respondents to ensure consistent interpretation of the terminologies and questions in the questionnaire. In general, respondents took no longer than thirty minutes to complete the questionnaire. Confidentiality and privacy were assured to all individuals returning the questionnaire and the organization that they represented.

2.4 Sample Selection

Our survey targeted the population at the organizational level (or alternatively at departmental level if there was more than one department in an organization responsible for software development). A draft questionnaire of the survey was trialed against a small group of five organizations, and a number of adjustments were made based on the experiences and feedback we gathered from the pilot run. As a result, we aimed at targeting four different types of participants in this survey. The first preference was test managers, the second was a member of the test team, thirdly a software project manager, and finally a general organizational or departmental manager. This allowed us to deal with situations where there was no specific individual responsible for testing in the organization.

Five approaches to our target audiences were made over a twelve month period to identify a suitable sample for the survey. Resources used were:- an article which appeared in the May 2002 issue of Australian Computer Society (ACS) magazine, *Information Age*, reaching around 14,000 Australian IT professionals [10]; a one-page insert in the February/March 2003 issue of *Information Age*; a list of 350 companies constructed from Australia's national telephone directories; a list of software test-likely organizations from classified post advertisements appeared in a newspaper[3]; and a flyer to request for participation enclosed in the June 2003 issue of the *Software* magazine published by Software Engineering Australia (SEA) with circulation of over 6,000 copies distributed to its members nationally.

As a result, a total of 65 individuals or companies participated in the survey. This is a relatively low response rate, given the large number of organizations that were invited to participate in the survey, and the large estimated size of the population.

During the pilot study of the survey, a "focus groups" sampling method was used, in which we personally invited companies that survey members had connections with to participate.

This survey sample was then built in three stages. In the first round, non-probabilistic sampling called "convenience sampling" [4] was employed, where the participants were selected because they were easy to access or because we believed they had a good chance of representing the population. In the mail out stage, "cluster based sampling" [4] was adopted, in which the target population was filtered using an indicator that was deemed likely to classify them as not being a software test-likely organization. Companies which had shop fronts and software/hardware sales companies were considered unlikely to be software development organizations and hence were unlikely to be performing any software testing. Nevertheless, the response rates in all data-collection stages of the project were far below our expected target of 100 responses or more, although based on our conversations with other researchers, this reflects the experience of others in Australia attempting to gather similar information in different disciplines.

The relevance of the sample was, however, considered to be extremely important, in that over 70% of respondents had managerial or team leadership roles in their organization and we are satisfied that the results from the sample are likely to be "indicative", although may not be absolutely conclusive. In particular, the results show the attributes of those responding organizations, regardless of whether they perform software testing in an ad hoc or a systematic manner.

Discussions among university colleagues have suggested that the low response rate may indicate that a large number of software development groups do not use any vigorous testing methods. It is also possible that the Australian software developers, similar to their New Zealand counterparts [3], are "survey averse", and that the cost of attaining representative samples is beyond the scope of our current project budget. Nevertheless, we intend to investigate the reasons why practitioners were reluctant to participate in the survey as part of our follow-up activities of the project.

3. Survey Results

3.1 Organization Information

[2] URL of the software testing survey web site is http://acssesurvey.it.swin.edu.au

[3] This task is simplified by the fact the largest circulation newspapers run extensive IT supplements on Tuesdays of each week.

Of the 65 organizations responded to our survey, more than two-thirds (67.7%) belong to the local private commercial sector. In addition to these, 15.4% were from overseas-based private commercial organizations, 10.8% were from government, and 6.2% were public non-commercial organizations (Table I).

Table I - Respondents by sector

Sector Type	Response	%
Government	7	10.8
Public non-commercial organization	4	6.2
Local private commercial organization	44	67.7
Overseas-based private commercial organization	10	15.4
Joint venture between public and private sectors	0	0
Total	65	100.0

The majority industry type of the respondent organizations was software house and IT consultancies (49.2%). Other industries included finance and insurance, manufacturing and engineering, research and development, and telecommunications (Table II).

Table II - Respondents by industry

Industry Type	Response	%
Banking, finance & insurance	7	10.8
Education & training	1	1.5
Hotel, tourism, retail & trading	2	3.1
Manufacturing & engineering	4	6.2
Research & development	3	4.6
Software house & IT consultancy	32	49.2
Telecommunications	3	4.6
Other	13	20.0
Total	65	100.0

The 65 organizations ranged from large in size with over 500 employees (24.6% of organizations) to very small sizes of less than 20 (also 24.6%). Most of these had substantial experience in software development: 13 organizations claimed to have 6 to 10 years of relevant software testing experience, 19 organizations with 11 to 19 years, and 22 organizations have more than 20 years of experience. Again, although the survey sample size was not ideal, we believe that these 65 organizations of such diversities do provide us a valid set of sample data and allow us to reflect the current software testing practices in the country.

As our main interests are in software testing, our questions mainly focused on software testing issues, including budget allocation. We found that only 3 out of 65 organizations allocated 40% or more of the total development budget to software testing in the initial software development plan, while 49 organizations had allocated less than 40% of the budget to testing. Among these 49 organizations, most of them (16 each) allocated between 10 to 19% or between 20 to 29% of the initial budget to testing alone. Nine organizations allocated between 30 to 39%, and amazingly there were 8 organizations which allocated less than 10% of the total development budget to software testing during the planning phase. Despite these facts, only 11 organizations (16.9%) reported that they met their testing budget estimates. Twenty-seven organizations (41.5%) spent 1.5 times of the estimated cost in testing and 10 organizations (15.4%) even reported a ratio of actual to estimated testing cost of 2 or above. Even more surprisingly, there were 3 organizations (4.6%) which managed to complete testing activities using only half of their initial allocated testing budget.

3.2 External Consultants, Testing Responsibility and Organizational Issues

We were surprised to find that, in the past 3 years, 24 organizations (36.9%) hired external testers to assist the organization to implement software testing methods or tools. Of these, 50% outsourced less than 20% of the testing budget to external testers, and 29.2% outsourced between 20 to 39%. In terms of satisfaction level, 75% were either satisfied or highly satisfied with the service from external testers and another 16.7% were neutral. Only 1 organization (4.2%) was dissatisfied and one other was highly dissatisfied. These figures clearly indicate that current external software testing companies are providing a very high standard of services to their clients in Australia.

The majority of respondents (70.8%) were found to appoint a person who is solely responsible for managing software testing activities in their organization, showing that testing is becoming a more independent process in industry.

User acceptance testing and regression testing were extremely common for all software applications developed, the results being 31 (47.7%) and 45 organizations (69.2%) respectively. Of the 45 organizations performing regression testing, 24 of them (53.3%) repeated regression testing for every new version of the application whilst 13 organizations (28.9%) conducted regression testing again after every change in the application.

Another interesting finding was that 50 out of 65 organizations (76.9%) followed formal processes or procedures for approving changes in requirements and specifications during the software development lifecycle. There were also 50 out of 65 organizations that formally documented requirement and specification changes during system development. In other words, the remaining 15 organizations (23%) did not formally document these changes at all. A closer scrutiny of the raw survey data indicated that there was no significant correlation between the 50 organizations in which formal processes were followed to approve

requirement changes and those 50 organizations in which requirement changes were formally documented during system development. This observation reveals the existence of some degrees of inconsistencies and weaknesses within the software development practices in industry.

3.3 Software Testing Methodologies and Techniques

This section investigated the extent of adoption of software testing methodologies and techniques in organizations to improve the quality of their software products. Forty-two out of 65 organizations (64.6%) claimed the use of at least one structured software testing methodology in the past 3 years. While it is encouraging to see that almost two-thirds of the respondents employ some structured testing methodologies, the fact that slightly more than one-third of the organizations are still doing ad-hoc testing was quite remarkable. In fact, we imagine that the actual figures for ad-hoc testing may be even underestimated, as many such organizations may be reluctant or may not have been interested in responding to our survey.

The three most popular methodologies included test case selection, static analysis and dynamic analysis. In terms of selecting test cases, black-box testing (particularly boundary value analysis and random testing) were more common than white-box testing (29 responses for black-box versus 16 for white-box). Eighteen respondents adopted data flow analysis techniques. Only 3 organizations reported the use of mutation analysis and none reportedly use symbolic analysis. Although the unpopularity of such techniques may not be conclusive due to the small sample size in the survey, it is evident that these techniques are rarely used in the industry despite large volume of research work has recently been done in these areas [1, 2, 6, 7]. Comparing static analysis and dynamic analysis, we observed that document and code inspection attracted a slightly higher response rate than code walkthroughs (29 versus 22). In both cases, manual processes were still more commonly engaged than automated ones (. The use of automated tools in software testing will be further discussed in later sections.

Of the 42 organizations using some form of structured testing methodology in the past three years, 27 (64.3%) carried out structured testing for more than 80% of their projects, and 21 organizations (50%) have been adopting structured testing methodologies for over 5 years. While 10 respondents (23.8%) were unsure if the cost-effectiveness has been improved by the use of methodologies, 28 (66.7%) expressed their affirmative responses, in contrast to only 4 respondents (9.5%) who expressed their disappointment in adopting testing methodologies. It would be interesting to further investigate the reasons why there exists such a large percentage of respondents who were unsure about the effects of utilizing systematic testing approaches.

Major testing activities performed by respondents (in order of popularity) were designing test cases (55 organizations), documenting test results (54 organizations), re-using the same test cases after changes were made to the software (also 54 organizations), defining test objectives (48 organizations) and re-designing test cases based on the analysis of previous test results (41 organizations). We observed that although some organizations did not claim to use structured testing methodologies, they did perform basic testing activities such as designing test cases and documenting test results on a regular basis.

Among the 56 organizations (86.2%) that used standard test plan templates, 18 of them (32.1%) *always* updated the test plan whenever there was a change in requirements and specifications. While 22 (39.3%) and 12 (21.4%) organizations respectively *quite often* and *occasionally* updated their test plans, surprisingly, 4 organizations (7.1%) *never* update their test plans, even when requirements and specifications changes occur. This survey result suggests that some organizations still may not be practicing the proper procedures of continuously updating test plans even though this process is generally regarded as essential to guarantee the validity and efficiency of test plans.

There were 59 out of 65 organizations (90.8%) reporting that formal tests were performed to ensure the developed software meets its requirements and specifications, suggesting that user acceptance testing is widely used in industry. Twenty-five organizations (38.5%) reported that over 80% of their test cases generated in the past 3 years were derived from specifications, with 17 organizations (26.2%) reporting between 60 and 79%. Regarding the percentage of software faults detected in the past 3 years, 22 organizations (33.8%) found that between 40 to 59 % of such faults were related to specification defects, followed by 16 (24.6%) and 15 organizations (23.1%) falling within the range of 20 to 39% and 0 to 19% respectively.

As expected, “big bang” was the most popular integration testing approach (used by 33 organizations) probably due to its simplicity. This was followed by bottom-up and top-down approaches, which were used by 27 and 23 organizations respectively.

Pre-defined criteria were used by 48 respondents (73.8%) to stop testing of a software system. Face to face interviews revealed that several organizations still

adopt the common practice of ceasing testing once resources are exhausted, irrespective of possible number of faults that may remain in the software. Another common trend was to cease testing as soon as all "critical" or "show-stopper" faults had been detected and removed, despite the fact that those methods used to determine whether all such faults had been removed were, in most cases, neither formal nor methodological in nature.

Being software testing researchers, we were particularly interested in practitioner's views on the barriers to adopting testing methodologies in their workplace. The responses from the survey were summarized in Table III.

Table III - Barriers to adoption of testing methodology

Barrier	Response	Rank
Do not think there is any barrier	20	2
Lack of expertise	**28**	**1**
Lack of support tools	18	
Costly to use	14	
Difficult to use	3	
Time-consuming to use	20	2
Do not think it is useful or cost effective	5	
Do not know of any testing methodology	7	
Other	14	

As indicated in Table III, 43.1% reported that a lack of expertise as the dominant factor preventing or disadvantaging organizations from using software testing methodologies. About 30% of respondents did not believe there was any barrier to using methodologies in their organizations. On the other hand, the same number of respondents regarded testing methodologies as being time-consuming when used.

The largest problem reported with using testing methodologies was a lack of expertise, with almost half of the respondents encountered. In our opinion, there are two likely causes of this. Firstly, this could indicate that software testing professionals are not sufficiently trained in testing methodologies either at the university or industry level. The second cause may be that there is a genuine shortage of software testing professionals with such knowledge in industry. In either case, it is obvious that training opportunities of software testers are essential to improve the quality and reliability of the software products developed in the country.

3.4 Automated Software Testing Tools

There was substantial usage of automated software testing tools amongst the respondents. In the past 3 years, 44 organizations (67.7%) have automated some of their testing activities. Out of these 44 organizations, 30 (68.2%) acquired the tools by purchasing existing commercial products, while only 6 (13.6%) developed their own tools. Quite unexpectedly, we found that only 1 organization (2.3%) out-sourced development of their testing tools.

Among these 44 organizations, automated testing tools for test execution (35 organizations or 79.5%), regression testing (33 organizations or 75%), and test results analysis and reporting (27 organizations or 61%) ranked the top three positions for automated testing activities. Other activities such as generating test cases/scripts and test planning/management also attracted more than one-third of the respondents (20 and 17 organizations respectively). A large proportion of respondents (36 organizations or 81.8%) in fact employed multiple automated techniques in software testing.

Although it is widely believed that software quality will be improved by the use of automated testing, only 30 of the 44 respondents (68.2%) using testing tools agreed with this belief. Ten organizations (22.7%) were unsure, and 4 organizations (9.1%) gave a negative response to this question.

About half (32) of the 65 respondents reported that cost was a major barrier to using automated tools for software testing in their organizations. There were 26 and 16 respondents respectively regarding time and difficulties as factors which prevented them from using testing tools in their organizations. The actual response figures were presented in Table IV.

Table IV - Barriers to adoption of testing tools

Barrier	Response	Rank
Do not think there is any barrier	9	
Costly to use	**32**	**1**
Difficult to use	16	3
Time-consuming to use	26	2
Do not think it is useful	4	
Do not think it is cost-effective	9	
No information resource available	1	
Do not know of any software testing tool	4	
Other	28	

3.5 Software Testing Metrics

Out of the 65 survey respondents, only 38 (58.5%) used measurable test objectives. Not surprisingly, the most popular metric reported was defect count (used by 31 organizations), probably due to its simplicity.

It is encouraging to see that 19 (50%) of the 38 organizations using metrics applied them to more than 80% of the software applications developed in the past 3 years. However, only 21 organizations (55.3%) agreed that the quality of the developed software applications was improved by the use of the metrics. Thirteen organizations (34.2%) were unsure, and 4 organizations (10.5%) even disagreed about the

positive effect of metrics on software quality. This counter-intuitive result certainly deserves further investigation as follow-up activities of the project.

As shown in Table V, about 30% of the participants (20 organizations) reported no barrier or disadvantage in the use of metrics. On the other hand, there was about a quarter of respondents (17 organizations) who found the use of metrics to be too time-consuming.

Table V - Barriers to adoption of testing metrics

Barrier	Response	Ranking
Do not think there is any barrier	**20**	**1**
Costly to use	4	
Difficult to use	4	
Time-consuming to use	17	2
Do not think it is useful	5	
Do not think it is cost-effective	2	
No information resource available	6	
Do not know of any software testing metrics	3	
Other	22	

3.6 Software Testing Standards

Software testing standards were being adopted by 47 out of 65 respondents (72.3%). In-house developed standards were employed by 29 organizations, while 18 organizations used a combination of published and in-house standards for software testing. Nevertheless, there were only 3 organizations relying solely on published standards, indicating that those standards known to software developers were possibly quite deficient. On the whole, 39 (83%) of the 47 organizations agreed that such standards did improve the software development processes used in their organization, none disagreed, and 7 were unsure (14.9%)[4].

Of the 65 organizations responded to the survey, 22 (33.8%) were quality accredited for their software development processes. Interestingly, out of these 22 accredited organizations, only 15 (68.2%) believed that their software development processes were being improved by acquiring the accreditation, while 4 (18.2%) did not think so and another 3 (13.6%) were not sure.

Table VI summarizes the respondents' views on the barriers to adopting software testing standards in their organizations. The majority of them (28 organizations) thought that there was no barrier. There were also significant numbers of responses indicating that time (15 organizations) and cost (13 organizations) are the other two main deterrents to the use of testing standards.

[4] One survey participant had mistakenly left this response blank, thus the total percentage in this category does not add up to 100.

Table VI - Barriers to adoption of standards

Barrier	Response	Rank
Do not think there is any barrier	**28**	**1**
Costly to use	13	3
Difficult to use	4	
Time-consuming to use	15	2
Do not think it is useful	6	
Do not think it is cost-effective	5	
No information resource available	3	
Do not know of any software testing standards	4	
Other	18	

3.7 Software Testing Training and Education

It was very encouraging to see that 47 (72.3%) out of the 65 responding organizations provided some opportunities for their software testing staff to receive training in software testing. Commercial external training courses were the most popular (reported by 37 organizations), followed by internal courses (25 organizations) and self-study (22 organizations). In terms of frequency of training, 28 (59.6%) out of the 47 organizations provided training to staff only on a needs basis. It is to our disappointment to report that only 7 organizations (14.9%) offered regular training to their software testing employees.

Table VII - Barriers to provide training to software testing staff

Barrier	Response	Rank
Do not think there is any barrier	18	3
Cost	**31**	**1**
Time	22	2
Course	14	
Other	10	

In terms of barriers to providing training, cost is still considered to be the most significant factor (31 organizations), followed by availability of time (22 organizations). It is indeed disappointing to see that there are only 18 (27.7%) out of 65 organizations that did not believe there was any barrier to provide training to software testing staff (Table VII).

3.8 Test Organization - Teams, Independent Testers and Training

Out of the 65 organizations, 44 of them (67.7%) had an independent testing team. Among these 44, 26 organizations (59.1%) had over 80% of independent testers in the software testing team (i.e. testing personnel that do not participate in any software design or implementation activities). Furthermore, only 10 organizations (22.7%) had over 80% of their testing team members completing formal training in software testing, and 7 organizations (15.9%) had 60 to 79%. However, there were also 15 organizations (34.1%)

with less than 20% of their testers being formally trained. As mentioned earlier, this finding reveals the inadequacy of formal training of many testing staff, and suggests that there may be an urgent need to provide more opportunities for formal training in software testing.

From the collected data, 27 participants (61.4%) reported that less than 20% of their testing team members received training in software testing through university studies. There were 9 organizations (20.5%) reported to have over 80% of their testing team members trained by in-service training courses, whilst in 7 organizations (15.9%) this was between 60 to 79%. However, as many as 19 organizations (43.2%) had less than 20% of their testers receiving formal training by attending in-service training courses. This high percentage may indicate that there is a possible divergence between the courses provided by commercial providers and the actual needs of the industry. In addition, when asked for their required minimum qualification for software testers, more than one-third of organizations specifically required candidates with some previous testing knowledge and experiences, indicating that there is a very high demand to offer more education and training opportunities to the novice software testers.

4. Analysis and Summary of Survey Findings

As stated by Kitchenham and Pfleeger in [4], if a sample is not representative of the population then one cannot make definite generalizations of the population. Therefore, due to the smaller than expected survey sample we were unable to prove or disprove our hypotheses. Nevertheless, the survey provides some very valuable insights to the current software testing practices in Australia. This section gives a broader analysis of our survey findings.

4.1 Major Barriers and Disadvantages

The most evident barrier to using software testing methodologies and techniques was found to be a lack of expertise among the practitioners, with almost half of the respondents giving the same answer. This finding suggests that there could be a vast number of software testing staff who are not been appropriately trained in the use of formal testing methodologies or techniques. This may signify a deficiency in the training of software testing professionals to meet the actual demand of the industry, or deficiencies in the techniques themselves.

Cost was ranked first in the list of barriers to the use of automated testing tools (Table IV) and also in the list of barriers to provide training to software testing staff (Table VII) in organizations. In fact, cost was also ranked highly as a barrier to using testing metrics and standards in organizations. This could possibly be due to impact of the IT economy downturn in recent years, resulting in a much more competitive environment in the current IT industry.

Time is another critical impediment in the view of respondents. A high proportion regarded using automated tools (Table IV), metrics (Table V) and standards (Table VI) in their organizations as time consuming.

Difficulty of use was ranked (by about one quarter of respondents) as the third barrier to adopting automated testing tools (Table IV). There could be three reasons for this. Firstly, organizations may not be familiar enough with automated tools in general, so when they intend to purchase a tool they have no way of assessing the type of tool they require or how to judge the ease of use of the tools. Conversely, it could be that tool vendors do not provide sufficient on-the-job training when selling their tools to organizations. Thirdly, the tools themselves may be difficult to adopt. The same factor was ranked fifth in the metrics section (Table V).

4.2 Organization Sectors Adopting Structured Testing Methodology

It is our initial feeling that Government and public non-commercial organizations, being public-funded, are very likely to adopt structured testing methodology. To our surprise, we found that while there are about 70% of private organizations (both local and overseas) adopting some form of structured testing methodology, Government and public organizations reported a significantly lower percentage. Although this observation is only indicative due to the small sample size, it does reveal there is substantial room for improvement in software testing practices within government and public organizations.

4.3 Popularity of the Test Case Derivation Methods

Section 3.3 shows that in general, test case derivation is reasonably widely used amongst the respondents. Our conjecture would be that this is a manual process that connects to some extent with design practices, and which may support demonstrations to users more readily than automated test case generation. It is also possible that existing undergraduate computer science and software engineering programs embed this in their programming and/or testing subjects. The survey results also reveal

that deriving test cases from specifications (i.e. using black-box strategies) was likely to be more popular than deriving test cases from program codes (white-box strategies) in industry.

4.4 Testing Budgets

As reported in Section 3.1, about three-quarters of organizations allocated less than 40% of their development budget to software testing activities and only about one-fifth of the organizations could adhere to or spend less than their allocated testing budget. This could be a strong indication that software development organizations are not allocating realistic budgets to testing, or that their methods of estimating testing costs are non-realistic. We encourage organizations to establish databases of both estimated and actual testing costs in various kinds of software development projects, thus providing real life data for more accurate estimation of testing costs in future projects. It could be interesting to further study the principal strategies adopted by organizations in allocating budget to testing in the planning phase of the projects.

4.5 External Testers

As an overall analysis, there was a substantial level of satisfaction among organizations that hired external testers to assist them in software testing activities. We predicted that in future years, hiring external testers may become even more popular. This certainly indicates an increasing need of professional testers in Australia. At the same time, certification of software testers may become progressively more important, in order to guarantee the standard of service offered by external testers.

4.6 Stopping Rules and Metrics

One major point of concern with the survey responses was the methods of deciding when to stop testing (Section 3.3). While there is a number of practitioners still using such rules, stopping when resources run out is not regarded as a reasonable metric [9]. However, defining and using stopping rules is never simple or easy. Without the use of statistical models such as fault seeding or confidence bounds as discussed by Pfleeger [8] or reliability models derived by Musa and Ackerman [5], it could be potentially risky and even disastrous to the quality of the software by using non-statistical criteria.

As a matter of interest, 35 out of the 42 organizations (83.3%) using structured testing methodologies also used stop-testing criteria, and 33 out of the 42 respondents (78.6%) using structured testing methodologies also used testing metrics. This could indicate that the majority of organizations in industry that use structured methodologies also use metrics or stop-testing criteria. This phenomenon is further reinforced by the observation that 30 out of these 42 organizations (71.4%) employ both stop-testing criterions as well as metrics. It is also interesting to observe that out of the 22 organizations which do not use any structured testing methodology, 9 of them (40.9%) neither use software testing metrics nor stop-testing criterions at all. It seems fair to say there still exists a significant fraction of practitioners performing ad-hoc testing activities in Australia.

4.7 Automated Tools

As reported in Section 3.4, the most popular type of tools used is to support test execution (35 out of 44 organizations), followed by regression testing (33 organizations), with result analysis and reporting tools (27 organizations) being the third. This result is not surprising to us as these activities are very labour intensive and as such there are plenty of well-established tools in the market to handle these tasks.

Another interesting point to report is that out of the 42 organizations that use structured testing methodologies, 34 (80.9%) also used automated tools, while 10 out of 23 (43.5%) did not use any testing methodology but did use testing tools. These results show that there exists a large demand of automated tools in the software testing industry. Provided that these tools are of high quality and the tool vendors provide sufficient training to the users, organizations are eager to adopt automated tools to facilitate their testing activities.

4.8 Standards

As reported in Section 3.6, very few organizations reportedly used published standards. Most organizations that use standards either develop their own from scratch or modify published standards to suit their needs. This insinuates that there may be a deficiency in the existing published software testing standards to suit the environment of Australian's organizations, and suggests that relevant professional bodies in Australia, such as ACS or SEA, should consider forming a special interest group to establish a set of software testing guidelines specifically for practitioners in Australia, and then transform these guidelines into standards when they are further improved and generally accepted by the majority of practitioners in Australia.

4.9 Training and Education

Our results indicate that training courses offered by universities or TAFE colleges contribute only 10.7% of the total training opportunities for organizations to train their testing staff. This may be due to the lack of practical skills delivered to tertiary students in traditional software testing courses, or the lack of short courses in software testing at university. We anticipate that in the future, more practical research in software testing will be carried out in universities. Perhaps these research results could be incorporated into university courses to provide more modern and useful skills to students and meet the rising demands of industry.

5. Conclusions and Future Work

In this paper, we presented and analyzed the findings of our preliminary software testing survey conducted in several major capital cities in Australia between 2002 and 2003. Although the sample size was smaller than ideal, we are confident that our findings reveal some trends of the current industry practices in software testing.

As a second stage of the survey, we plan to increase the sample population to facilitate a more vigorous statistical analysis of the obtained data. We would also like to compare the data from the Australian industry to that obtained from other Southeast Asian countries in order to assess the competitiveness of Australia among its neighbours in the Asia-Pacific region.

Since the reliability of the survey sample has not been firmly established, all organizations involved in the mail out that did not respond to our call for participation will be contacted again to ascertain their reason for not participating. This may give a better indication as to the reliability of the survey sample, as well as whether or not our generalizations are valid.

As we remarked earlier, there is anecdotal evidence in Australia at least, that substantial resources are being committed to testing by some developers. At the same time, users continue to grapple with faulty software (at a time when extremely high quality infrastructure systems, e.g. EFTPOS, exist).

The need for surveys of this type is clear. Establishing the optimum relationship between testing and software quality; that is ensuring that testing strategies are in place which yield the highest quality software, is increasingly important as software begins to intrude more and more into people's daily lives. We are convinced that this survey, despite its limitations, will assist in this process.

Acknowledgements

The work described in this paper was supported by a grant from the Australian Computer Society. We would also like to acknowledge Australian Computer Society and Software Engineering Australia for including our promotion flyers in the magazines for circulation to their members. Dr. Marcelle Schwartz of La Trobe University provided invaluable statistical advice, and Thilky Perera of Bond University assisted in developing the survey sample. Last but not the least, we are grateful to all respondents of the survey. Without their efforts and enthusiasms, this survey could never be successful. Finally, any errors or omissions are the fault of the authors.

References

[1] P. Ammann, "System Testing via Mutation Analysis of Model Checking Specifications", *ACM SIGSOFT Software Engineering Notes*, Volume 25, No. 1, January 2000, pp 30.

[2] Murial Daran, Pascale Thévenod-Fosse, "Software Error Analysis: a Real Case Study involving Real Faults and Mutations", *ACM SIGSOFT Software Engineering Notes*, Volume 21, No. 3, May 1996, pp 168-171.

[3] L. Groves, R. Nickson, G. Reeves, S. Reeves and M. Utting "A Survey of Software Practices in the New Zealand Software Industry", *Proceedings of the.2000 Australian Software Engineering Conference,* Gold Coast, Queensland, Australia, April 28-30, 2000 pp 189-101.

[4] B. Kitchenham and S. L. Pfleeger, "Principles of Survey Research, Part 5: Populations and Samples", *ACM SIGSOFT, Software Engineering Notes*, Volume 27, No. 5, September 2000, pp 17-20.

[5] J. D. Musa and A. F. Ackerman, "Quantifying Software Validation: When to Stop Testing?", *IEEE Software*, May 1989, pp 19–27.

[6] N. Mateev, V. Menon, K. Pingali, "Fractal Symbolic Analysis", *Proceedings of the 15th international conference on Supercomputing*, June 2001, pp 38-49.

[7] T. Murnane and K. Reed. "On the Effectiveness of Mutation Analysis as a Black-box Testing Technique", *Proceedings of the 2001 Australian Software Engineering Conference*, Canberra, Australia, 2001, pp 12-20.

[8] S. L. Pfleeger, *Software Engineering: Theory and Practice*, Prentice-Hall, USA, 2001.

[9] R. S. Pressman, *Software Engineering: A Practitioners Approach*, McGraw-Hill, International Edition, 1992.

[10] K. Reed, "Testing, testing, testing", *Australian Computer Society Information Age*, April/May 2002, pp 56–58.