Journal of Theoretical and Applied Information Technology
15th December 2016. Vol.94. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SOFTWARE CONFIGURATION MANAGEMENT PRACTICE IN MALAYSIA

[1]SYAHRUL FAHMY, [2]AZIZ DERAMAN, [3]JAMAIAH H. YAHAYA

[1]Faculty of Computer, Media & Technology Management, TATI University College, MALAYSIA

[2]School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu, MALAYSIA

[3]Faculty of Information Science and Technolgy, Universiti Kebangsaan Malaysia, MALAYSIA

E-mail: [1]fahmy@tatiuc.edu.my, [2]a.d@umt.edu.my, [3]jhy@ukm.edu.my

ABSTRACT

This study investigates the practice of Software Configuration Management (SCM) in Malaysia. A survey was carried out involving three types of software organizations namely the Government, IT Companies and Institutes of Higher Learning to identify the approach for SCM implementation the aspects of: Process, People and Documentation. Results revealed that Malaysian organizations are adopting key SCM Process where versioning of artifacts and releasing of baselines are highly exercised; policy and procedures for change control exist; audits are carried out periodically; and software libraries are used in software projects. In terms of tools, commercial tools are dominantly used which include Serena Dimensions, Borland Calibre and Oracle Discoverer. Documentation efforts are carried out comprising of internal and external conformances; software quality factors; controlled artifacts; and audit results. Main project reports generated include software development, audits and change requests. Among the issues observed are the exclusion of some artifacts as controlled artifacts; the lack of effort to document software quality evaluation techniques; and little effort to include SCM and software quality documentations in software delivery.

Keywords: *Software Engineering, Software Configuration Management, Software Quality*

1. INTRODUCTION

The Malaysian *Information and Communication Technology* market has strong growth fundamentals with key sectors that include the government, telecommunications, and education. Malaysia has more than 20 million Internet users (2015 statistics), and as such, the demand for enabling technologies for Malaysia's 1.1 million registered businesses into the new economy are high. The *Malaysia Digital Economy Corporation* (MDEC) is tasked to drive the nation's digital economy agenda, capitalizing on the 3,800 tech companies and 150,000 knowledge workers across the country [1].

The *National Transformation Programme* is undertaken to stimulate the development of the nation's digital economy by 2020. This programme will create an ecosystem that promotes the use of ICT in all aspects of the economy to increase the Gross National Income. The contribution of digital economy towards the Gross Domestic Product rose from 16.38% in 2013 to 17% in 2014, resulting in over RM295 billion in revenue to the Malaysian economy, RM283 billion worth of investments, and more than 147,000 jobs since 1996 [2].

The Malaysian Government is aggressively promoting the use of ICT in all industries with the development of critical software applications such as customer relations management, enterprise resource planning, human resource management, and financial management. In order to ensure timely delivery of such applications, it is crucial that a formal approach for *Software Configuration Management* (SCM) is adapted and appropriate supporting tools are used.

SCM is an area in software engineering that tracks and controls changes in the software life-cycle. SCM ensures that the current design and build state of a software product is known. Accurate historical record is useful not only for management and audit purposes, but also for development activities. Effective use of SCM enables developers to formally record all change details. One of the strength of SCM is the ability to rebuild software applications in the event of failure.

This study aims to investigate SCM implementation in Malaysia from the aspects of *Process*, *People* and *Documentation*. This paper is organized as follows: a brief review of SCM will be

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

presented in Section 2. Section 3 presents the tool used in this study and Section 4 presents the results of the survey carried out. Finally, discussion and conclusion will be made in Section 5.

2. SOFTWARE CONFIGURATION MANAGEMENT

SCM is a discipline in software engineering for controlling and managing changes to software products using standard processes and tools. IEEE [3] defines SCM as:

> *"a supporting-software life cycle process that benefits project management, development and maintenance activities, quality assurance activities, as well as the customers and users of the end product".*

SCM is a '*supporting process*' in the software product life-cycle. It support '*primary processes*' such as development and maintenance, and carried out continuously throughout the life-cycle.

Typical SCM implementation comprises of *Process*, *Tools* and *Documentation* (Fig. 1).

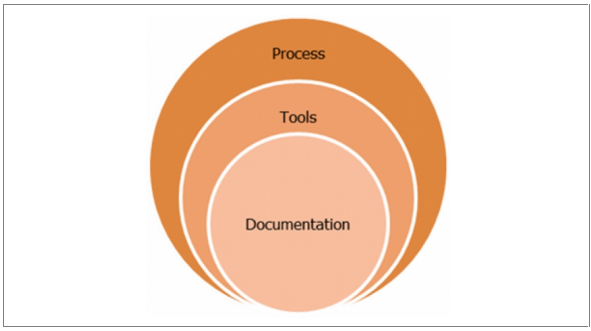

Fig. 1 Traditional SCM Model

Process refers to SCM policy and procedures that are adapted by organizations in software projects. *Tools* are used to support SCM implementation. Organizations usually use a set of tools, as opposed to a single tool, for SCM purposes. *Documentation* provides traceability and proof of conformance to requirements and standards. *Process* determines the type of *Tools* to be used and the type of *Documentation* to be generated.

2.1 Process

There are 3 major views on the process involved in SCM from the *Institute of Electrical and Electronics Engineers* (IEEE) [3], the *Software Engineering Institute* (SEI) [4] and the *International Organization for Standardization* (ISO) [5]. IEEE stipulates 6 processes in SCM; SEI outlines 7; while ISO specifies 5 processes. Although the number of process varies, the outcomes are similar as illustrated in Table 1.

Table 1: Comparison of SCM Process and Outcomes

IEEE	SEI	ISO	OUTCOME
1. Management and Planning of the SCM Process		1. Configuration Management Planning	Software Configuration Management Plan
2. Software Configuration Identification	1. Identify Configuration Items	2. Configuration Identification	List of items to be controlled
3. Software Configuration Control	2. Establish a Configuration Management System 3. Create or Release Baselines	3. Change Control	Policy and procedure for change process
4. Software Configuration Status Accounting	4. Track Change Requests 5. Control Configuration Items 6. Establish Configuration Management Records	4. Configuration Status Accounting	Record/ Reports on the approved configuration identification; status of new release and configuration; implementation of change, etc.
5. Software Configuration Auditing	7. Perform Configuration Audits	5. Configuration Audit	Formal and informal audit reports. Pre-requisite for the establishment of product baseline
6. Software Release Management and Delivery			Packaging and delivery of the elements of a software product

2.2 Tools

There is a wealth of tools for supporting the SCM process found in the industry (commercially) and literature (theoretically). Some of the notable tools in SCM include the *Source Code Control System* [6]; *Revision Control System* [7]; *Adele* [8]; *ClearCase* [9]; *SourceSafe* [10]; *SunForte* [11]; *Subversion* [12]; *Make* [13]; and *CCM* [14]. Tools that are applicable to SCM based on the processes identified in the previous section can be classified into 4 categories: *Versioning*; *Conflict Detection*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and Resolution; *Change Tracking*; and *Software Build*.

2.2.1 Versioning

Versioning tools are used to maintain software artifacts including source codes, configuration files and documentation. Changes are identified by a number or letter (revision number) and each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored, and merged, depending on the type of files involved. Examples version tools include *Gitless*, a version control system that covers common Git use cases [15]; *Phoca*, a system to support to fine-grained and flexible version control [16]; *EMFStore*, for versioning models based on the Eclipse Modeling Framework [17]; *Performance-Aware Revision Control Support* (PARCS), a service that provides feedback to developers how a change that they have committed affects the behavior and performance of the overall application [18]; and *Odyssey-VCS* for fine-grained UML model elements [19].

2.2.2 Conflict Detection and Resolution

Conflict detection and resolution tools are used to manage changes in a collaborative development environment were concurrent access to artifacts are made. Examples of these type of tools include *FSTMERGE* for semi-structured merge [20]; *WeCode*, a client and server plug-in, to automate the detection of conflicts as they emerge [21]; *Palantir*, a workspace awareness tool for SCM to assist developers in detecting and resolving conflicts [22]; *CASI*, a tool that informs developers of changes that are taking place in a software project and the source code influenced by them [23]; and *Semantic Conflict Analyzer* (SCA) to detect semantic interference between parallel changes [24]. Branching and merging strategies are usually taken into consideration when using these tools.

2.2.3 Change Tracking

Change tracking tools are used to monitor the project status and implementation progress. Depending on the type of artifacts, normal project management tools can also be used as tracking tools i.e. word documents or excel workbook. Tracking technical changes would require specialized tools for example tracking changes to single line of code as opposed to entire files [25]; detecting changes between versions of source code [26]; mechanism for history-based changes [27]; personalizing change notification feed [28]; expressing fine-grained changes [29]; scoping change impact analysis technique [30]; and automatically identifying a minimal number of code modifications across revisions [31].

2.2.4 Building

Software building tools are used to manage the selection and compilation of source codes into standalone software artifact(s). There are a number of commercial tools for supporting this process including *Make* (including derivations), *Apache Ant* (non-Make-based), *GNU Build* system (script-based), and *Microsoft's Team Foundation Server* (integration tool). Specialized tools include the integration of security checks to enhance system's usability and software quality [32]; and definition of reusable, parameterized and interconnected builders [33].

In addition to these tools, there are also SCM tools based on the type of software project for example SCM for *Software Product Line* [34]; for *Model-Driven Engineering* through MOD2 SCM, a configuration tool based on features from a comprehensive feature model [35]; for *Component-Based Systems* through a framework for managing life-cycle evolution of heterogeneous component systems [36]; for *Open-Source Systems* through TRICA, an open source framework consisting of Subversion, Trac, and Hudson [37]; for *Unified Models* through Sysiphus, a tool for collaborating over software engineering artifacts following the RUSE model [38]; for *Clone Management* through Clever, a clone-aware SCM system [39]; and for the *Cloud* through Cored, a collaborative development environment using social media [40].

2.3 Documentation

The ultimate result of the SCM process is the SCM Plan (SCMP), a *"living document"* which serves as a reference for the project and updated throughout the software life-cycle [3,41]. A typical SCMP would include 4 types of documentation: *Management*, *Activities*, *Schedules* and *Resources*.

2.3.1 SCM Management

SCM Management explicitly defines, among other things, the organization, responsibilities, authorities, policies and procedures that need to be observed throughout the SCM project. Examples of documentation include SCM organization chart, life-cycle process, tool selection, and branching/ merging strategies.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2.3.2 SCM Activities

SCM activities record the progress as well as executive decisions made throughout the project. These include the identification and selection of configuration items; the Change Control Board; change implementation status; audits; software building and others.

2.3.3 SCM Schedules

SCM Schedules coordinates activities and records key milestones in the project. Examples include version and baselines releases, audits, and change implementation.

2.3.4 SCM Resources

SCM Resources records the use of resources throughout the project including tools, physical assets, and human capitals.

2.4 Status of SCM

SCM can be seen as a mature area in software engineering with a solid theoretical foundation and a wealth of professional tools to support its implementation. The *Process* is well established and documented in international standards whilst *Tools* are commercially available. *Documentations* too are evolving based on the process adapted as reported by the industry and academia.

With that in mind, this study is interested to answer the material question of "*does the practice of SCM (in the industry) agrees with SCM best practices*?". Specifically, this study hopes to (1) identify the *Process* involved in SCM; (2) to identify the tools used to support SCM implementation; and (3) to identify of the types of *Documentation* used in software projects in Malaysia.

3. RESEARCH TOOL

Questionnaire was selected as the tool for this study. The questionnaire was sent to group of 4 experts representing the target respondents (Government, IT Company, IHLs), asking them to evaluate the suitability, consistency of term, and the measurement used in the questionnaire. These experts include software development team senior managers and technical group leaders in their organization. Two significant comments by this group were the language used and length of the questionnaire. The questionnaire was re-drafted, the length revised, and a Malay version was included.

3.1 Questionnaire

The final version has 40 questions in 6 sections: *Organization Background*; *Conformance*; *Control*; *Audit*; *Delivery*; and *Software Configuration Management* (Appendix 1).

3.1.1 Organization Background

This section identifies the background of respondents (organizations). There are 5 questions in this section and the information sought includes: type of organization; nature of business; number of employees; location of organization; and the number of software projects undertaken in the past 12 months.

3.1.2 Conformance

This section concerns the conformance issues faced by organizations in software projects. There are 7 questions in this section and the information sought includes: type of conformance requirements by external and internal stakeholders; type of software quality factors used in software projects; and documentation of conformance requirements and software quality factors.

3.1.3 Control

This section concerns the change control process in software projects. There are 9 questions in this section and the information sought includes: type of tools used in software projects; changes to software artifacts; procedure for change request; and documentation of project tools and artifacts.

3.1.4 Audit

This section concerns the audit process in software projects. There are 5 questions in this section and the information sought includes: type of audits; frequency of audits; software quality audits; and documentation of audit results.

3.1.5 Delivery

This section concerns the delivery of software. There are 6 questions in this section and the information sought includes: the use of software library; artifacts included in software delivery; number of overdue software projects; revised software projects due to software quality issues; and average completion time of overdue projects.

3.1.6 Software Configuration Management

This section concerns the versioning of artifacts and report generation in software projects. It also identifies the familiarity of respondents to SCM. There are 8 questions in this section and the information sought includes: versioning and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

baseline exercise; reporting tools; and the use of specific SCM tools.

3.2 Respondents

This study aims to get a general view of SCM practice in Malaysia and therefore, the *Government* (Public Sector), *IT Companies* (Private Sector), and *Institutes of Higher Learning* (IHLs) were selected as the respondents of the survey.

3.2.1 Government

This group is represented by the Federal Ministries of the Malaysian Government. The total number of ministries identified was 25.

3.2.2 IT Companies

This group is represented by software development companies registered with *The National ICT Association of Malaysia* (PIKOM). PIKOM has more than 1000 member companies, involved in a wide spectrum of ICT products and services, dominating 80% of the total ICT trade in Malaysia. The total number of companies identified was 139. These companies were directly involved in software development projects.

3.2.3 Institutes of Higher Learning

This group is represented by the Public and Private Institutions of Higher Education as listed by the Ministry of Higher Education. The total number of IHLs identified was 46.

3.3 Distribution

The survey was prepared using Google Forms and distributed by e-mail (Google Mail). E-mail addresses were obtained through the official websites of respondents. The questionnaires were accessible for 3 months before the links were disabled and responses were no longer accepted.

4. RESULTS

A total of 23 responses were received from all groups which is approximately 11.0% of the population. The highest return rate was in from the Government, followed by IHL and IT Companies at 12.0%, 10.9% and 10.8% respectively (Table 1).

Table 1: Survey Response Rate

GROUP	SENT	REPLIED	%
Government	25	3	12.0
IT Companies	139	15	10.8
IHL	46	5	10.9
TOTAL	210	23	11.0

4.1 Organization Background

The respondents that took part in the survey comprises of the Government (26%), IT Companies (58%), and IHLs (16%). IT Companies are divided into Public Limited Companies (16%), Private Limited Companies (37%) and Sole Proprietorship (5%) (Fig. 1).

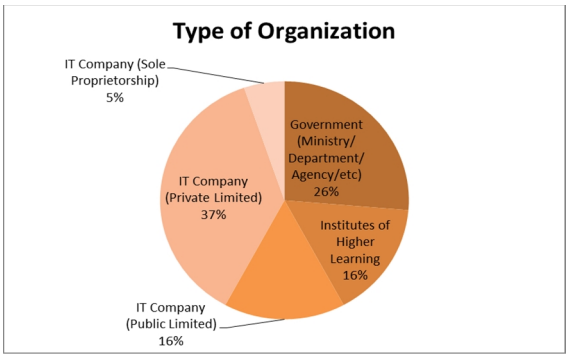

Fig. 1 Organization Type

42.1% of the respondents have less than 30 full-time employees, 36.8% have between 30 to 75, and 21.1% have more than 75. The respondents are involved in Communication & Networking (26.3%); Consultancy & Professional Services (36.8%); Creative Design/ Content (21.1%); Data Centre/ Web Hosting (21.1%); Distributor & Retailer (21.1%); Education & Training (26.3%); Internet-Based Business (5.3%); IT Outsourcing (10.5%); Maintenance (15.8%); Mobile & Wireless (31.6%); and Software Development/ System Integrator (89.5%) (Fig. 2).

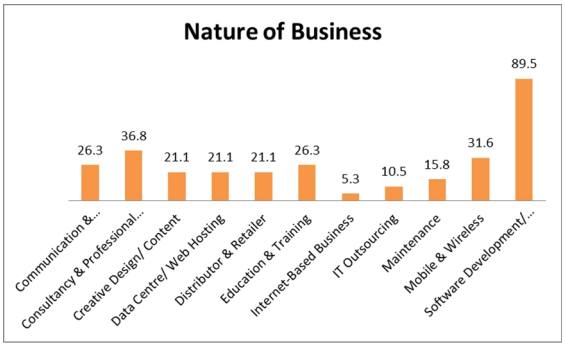

Fig. 2 Business Nature

Journal of Theoretical and Applied Information Technology
15th December 2016. Vol.94. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The average number of software projects undertaken in the last 12 months is 24. At the group level, the average number of software projects undertaken by the Government is 9, 22 projects by IT Companies and 37 projects by IHL.

4.2 Conformance

Conformance requirements imposed by internal and external stakeholders are as follows: Adherence to specific standards (68.4% and 57.9%); Use of specific resources (31.6% and 63.2%); Scheduling (94.7% and 84.2%); Software development strategies (15.8% and 73.7%); Software Configuration Management (10.5% and 57.9%); and Quality Assurance (84.2% and 89.5%) (Fig. 3).

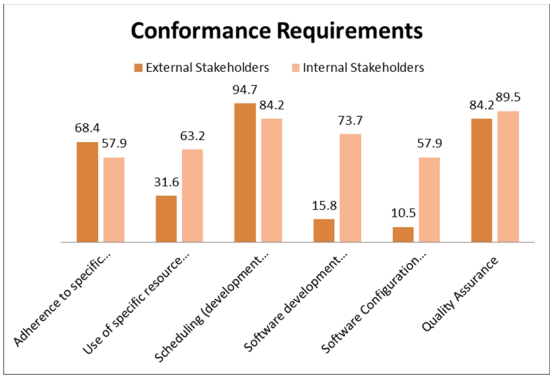

Fig. 3 Conformance Requirements in Software Projects

Internal and external conformances are documented by 94.7% of the respondents and 52.6% do not explicitly identify software quality factors in software projects. Quality factors that are frequently used in software projects are Functional Suitability (89%), Performance Efficiency (68.4%), Reliability (80.5%), and Security (84.7%) (Fig. 4).

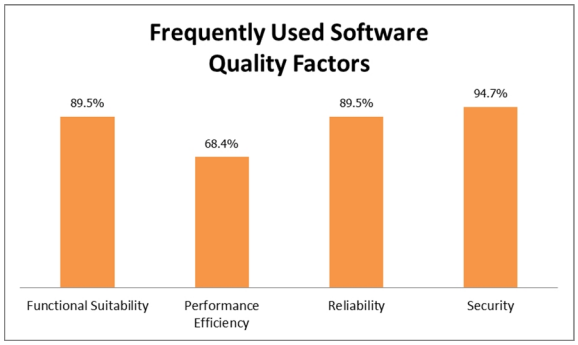

Fig. 4 Frequently Used Quality Factors in Software Projects

Less used quality factors are Compatibility (5.3%), Usability (5.3%), Maintainability (5.3%), and Portability (21.1%). Software Quality Factors are documented by 68.4% of the respondents but 73.7% does not document the techniques for assessing them.

4.3 Control

Software Development (100.0%), Software Configuration Management (57.9%), and Project Management (52.6%) are the most used tools in software projects. Project Reporting and Quality Assessment tools are used by 26.3% and 21.1% of the respondents; and Audit and Project Monitoring tools are used by 5.3% of the respondents (Fig. 5). The use of tools in software projects are documented by 78.9% of the respondents.

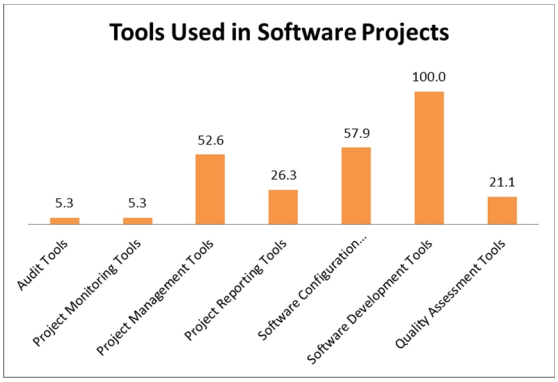

Fig. 5 Tools Used in Software Projects

89.5% of the respondents control changes to software artifacts particularly the Source Code (84.2%), Requirements Specifications Documents (78.9%), Tools (47.4%), Business/Process Logic (36.8), Product Documentation (31.6%), and Design Documents (10.5%). Audit (plans/procedures/results/etc.) is not controlled (Fig. 6).

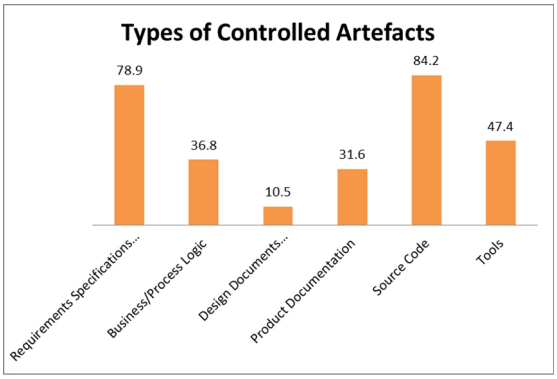

Fig. 6 Types of Controlled Artifacts in Software Projects

68.4% of the respondents reported that controlled artifacts are documented. The decision for artifacts control lies primarily with the Internal Stakeholders (73.7%). 94.7% have a specific policy and/or procedure for Change Requests which is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

communicated using a standard form (73.7%). The responsibility for accepting or rejecting Change Requests lies in the Project Manager (68.4%) and Other Internal Stakeholders (21.0%) (Fig. 7). Change Requests usually come from the Internal Stakeholders (63.2%).

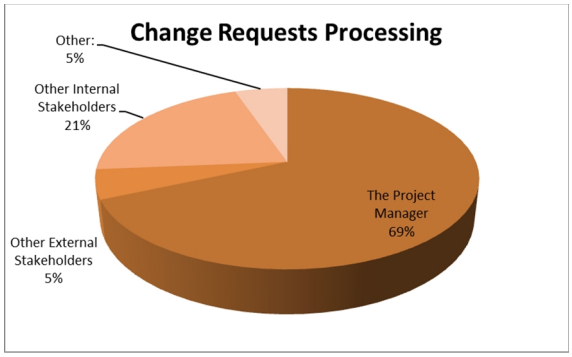

Fig. 7 Change Requests Processing

4.4 Audit

Informal Audits are carried out in all software projects, performed Periodically (47.4%) and As Needed (47.4%). The Internal Stakeholders determines the type of audit and when it will be carried out (84.2%). 57.9% of the respondents reported that a separate audit is carried out for software quality using standard test including Browser Compatibility Testing (63.2%), Load Testing (21.1%), Mobile Device Testing (52.6%), Network Testing (21.1%), Penetration Testing (26.3%), Security Assessment (21.1%), Vulnerability Scan (26.3%), Walkthrough (26.3%), and Others (42.1%) (Fig. 8). Results of audits are documented by 68.4% of the respondents.

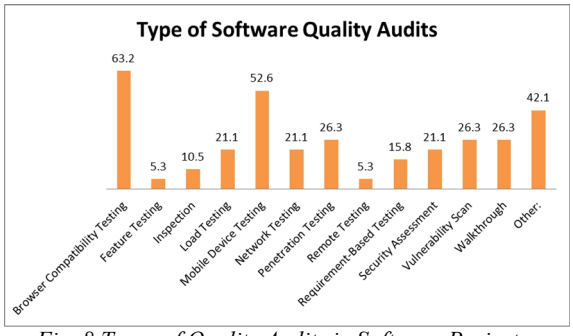

Fig. 8 Types of Quality Audits in Software Projects

4.5 Delivery

84.2% of the respondents use software library or central repository in software projects and 57.9% can reproduce previous versions of a software product easily. Among the artifacts that are included in software delivery include Product Documentation (89.5%), Executable Program (73.7%), Release Notes (73.7%), and Change Request Documentation (52.6%) (Fig. 9).

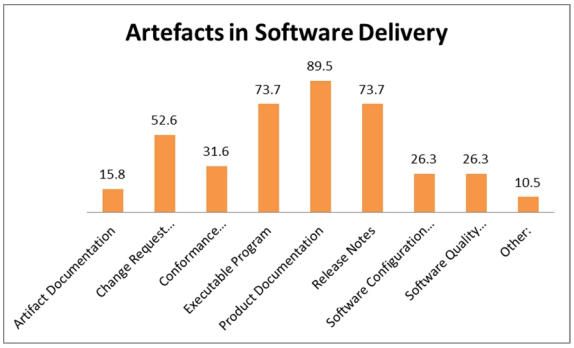

Fig. 9 Artifacts in Software Delivery

Overall, an average of 3 software projects could not be delivered on time; 3 projects were revised due to software quality issues; and average time for completion (of overdue project) is 2.3 months. Group averages are 1 overdue project, 1 revised project and 2.7 months completion time for Government; 3 overdue projects, 3 revised projects and 2.0 months completion time for IT Companies; and 4 overdue projects, 4 revised projects and 2.6 months completion time for IHLs.

4.6 Software Configuration Management

Versioning of artifacts are practiced by 94.7% of the respondents while 78.9% releases baselines in software projects. Reports that are used in software projects include Artifacts Report (21.1%), Audit Report (47.4%), Change Request Report (31.6%), Conformance Report (10.5%), Software Development Report (84.2%), Software Quality Assurance Report (21.1%), Tools Report (21.1%), and Others (10.5%) (Fig. 10). 57.9% of the respondents reported that they do not use reporting tools in software projects.

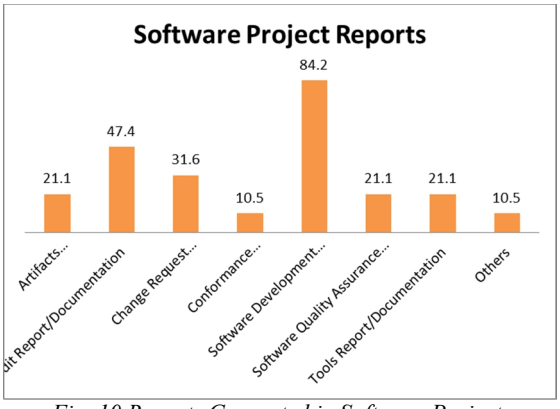

Fig. 10 Reports Generated in Software Projects

78.9% of the respondents are familiar with the term "*Software Configuration Management*" and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

68.4% adopt specific SCM approaches and tools. 89.5% agree that SCM could improve the quality of software products and the management of software projects in their organization.

The results paint a generalized view of software organizations in Malaysia with *IT Companies* making up most of the industry, followed by *IHLs* and *Government*. Apart from software development and system integration activities, these organizations are also involved in *Consultancy & Professional Services*; *Mobile & Wireless*; *Communication & Networking*; and *Education & Training*. Only a small percentage of the respondents are involved in *Maintenance*. The average number of software projects undertaken is 24 and the percentage of overdue projects is between 10.8% and 11.3%. Average completion time for overdue projects is 2.3 months. *IHLs* have the highest number of overdue projects which requires an additional 2.6 months for completion. Majority of the respondents agree that that SCM could improve the quality of software products and the management of software projects in their organization.

5. DICUSSION AND CONCLUSION

This survey has shed light to the SCM practice in Malaysia. SCM implementation is quite strong where the approach for SCM is based on standardized process; SCM implementation is supported by commercial tools; and crucial documentation for project monitoring and conformance verification are generated throughout the project. Results indicate that Malaysia is in the right track for creating a digital economy and becoming a regional service hub with regards to the development of critical software applications.

5.1 Process

The SCM practice in Malaysia is sound with organizations adopting key SCM process. Versioning of artifacts and releasing of baselines are exercised; policy and procedures for change control exist and communicated using standardized forms; audits are carried out periodically and as needed; and software libraries are used in software projects. Decisions for artifacts control; accepting or rejecting change requests; and type of audit and schedule lies highly with the *Internal Stakeholders*. A slight issue is noted where *Design Documents* are not considered part of controlled artifacts. We argue that these types of document should be controlled in order to not re-inventing the wheel with prior designs, hence adding more time to the project.

5.2 Tools

The appreciation and use of SCM support tools are promising with the majority of the respondents utilizing commercial tools such as *Serena Dimensions*, *Borland Calibre*, *Oracle Discoverer* and *Microsoft Office*. Although there is a lack of specialized *Project Reporting* and *Project Monitoring* tools used in software projects, this can be justified by the use of integrated SCM tools for project reporting and monitoring purposes.

5.3 Documentation

The documentation efforts in software projects in general and in SCM in particular are extensively carried out comprising of internal and external conformances; software quality factors; controlled artifacts; and audit results. Main reports generated include software development, audits and change requests. There is a minor issue of software quality evaluation documentation. We argue that software quality factors and corresponding evaluation technique(s) should be dictated as early as possible to guide the project and as proof of conformance of the software product. We also note that there is only little effort to include SCM and software quality documentations in software delivery. We argue that these artifacts should be included in software delivery to (1) facilitate future enhancements to the software product; and (2) prove of conformance to specific quality standards.

This study has also revealed the possibility of other influencing factor(s) in successful SCM implementation. Although several organizations surveyed have sound SCM process, tools, and documentation, they too however, experience project delays and revisions. Perhaps a factor worth looking into is the role of *People* or *Human* in SCM.

We would like to note that some respondents have opted not to take the survey since it would reveal their SCM process and tools. Although is limits to response of the survey, we assume that a standardized SCM process and appropriate support tools are in use.

Directions for future research include the investigation of *Human* factor in SCM; quantitative study on the effects of SCM and software quality documentations in software delivery; and the role of controlled artifacts selection in ensuring timely delivery of software products.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REFRENCES:

[1] Malaysia Digital Economy Corporation http://mdec.my/ Last accessed Sept 2016.

[2] MSC Malaysia http://www.mscmalaysia.my/ Last accessed Sept 2016.

[3] Bourque, P., and Fairley, R.E. (2014). Guide to the Software Engineering Body of Knowledge (SWEBOK Version 3.0). IEEE Computer Society Press, Los Alamitos, CA, USA.

[4] CMMI Product Team. 2010. CMMI for Development Version 1.3. Software Engineering Institute.

[5] International Organization for Standardization. 2003. ISO 10007: Quality Management - Guidelines for Configuration Management.

[6] Rochkind, M. J. (1975). The Source Code Control System. IEEE Transactions on Software Engineering, 1(4): 364–370.

[7] Tichy, W.F. (1982). Design, Implementation, and Evaluation of a Revision Control System. In Proceedings of the 6th International Conference on Software Engineering (ICSE '82). IEEE Computer Society Press, Los Alamitos, CA, USA, 58-67.

[8] Estublier, J., Ghoul, S., & Krakowiak, S. (1984). Preliminary Experience with a Configuration Control System for Modular Programs. In Proceedings of the 1st ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments. ACM, New York, NY, USA, 149–156.

[9] Leblang, D.B. (1995). The CM Challenge: Configuration Management that Works. In Configuration Management, Walter F. Tichy (Ed.). Wiley Trends in Software Series, Vol. 2. John Wiley & Sons, Inc., New York, NY, USA 1-37.

[10] Microsoft. (2000). Sourcesafe Product Documentation, Microsoft, Inc., Seattle, WA.

[11] Sun/Forte. 2000. Teamware Product Documentation. Sun MicroSystems Inc, Mountain View, CA.

[12] Pilato, M. (2004), Version Control with Subversion. O'Reilly & Associates, Inc., Sebastopol, CA, USA.

[13] Feldman, S. I. (1979). Make - A program for Maintaining Computer Programs. Software, Practice and Experience, 9(3): 255–265.

[14] Wright, A. (1990). Requirements for a Modern CM System. CaseWare, Inc.

[15] de Rosso, S.P., and Jackson, D. (2013). What's Wrong with Git?: A Conceptual Design Analysis. In Proceedings of the 2013 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software (Onward! 2013). ACM, New York, NY, USA, 37-52.

[16] Junqueira, D.C., Bittar, T.J., and Fortes, R.P.M. (2008). A Fine-Grained and Flexible Version Control for Software Artifacts. In Proceedings of the 26th annual ACM international conference on Design of communication (SIGDOC '08). ACM, New York, NY, USA, 185-192.

[17] Koegel, M., and Helming, J. (2010). EMFStore: A Model Repository for EMF Models. In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2 (ICSE '10), Vol. 2. ACM, New York, NY, USA, 307-308.

[18] Mostafa, N., and Krintz, C. (2009). Tracking Performance Across Software Revisions. In Proceedings of the 7th International Conference on Principles and Practice of Programming in Java (PPPJ '09). ACM, New York, NY, USA, 162-171.

[19] Murta, L., Correa, C., Prudencio, J.G., and Werner, C. (2008). Towards Odyssey-VCS 2: Improvements over a UML-Based Version Control System. In Proceedings of the 2008 International Workshop on Comparison and Versioning of Software Models (CVSM '08). ACM, New York, NY, USA, 25-30.

[20] Apel, S., Liebig, J., Brandl, B., Lengauer, C., and Kastner, C. (2011). Semistructured Merge: Rethinking Merge in Revision Control Systems. In Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering (ESEC/FSE '11). ACM, New York, NY, USA, 190-200.

[21] Guimaraes, M.L., and Rito-Silva, A. (2010). Towards Real-Time Integration. In Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering (CHASE '10). ACM, New York, NY, USA, 56-63.

[22] Sarma, A., Redmiles, D., and van der Hoek, A. (2008). Empirical Evidence of the Benefits of Workspace Awareness in Software Configuration Management. In Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (SIGSOFT '08/FSE-16). ACM, New York, NY, USA, 113-123.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[23] Servant, F., Jones, J.A., and van der Hoek, A. (2010). CASI: Preventing Indirect Conflicts Through A Live Visualization. In Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering (CHASE '10). ACM, New York, NY, USA, 39-46.

[24] Shao, D., Khurshid, S., and Perry, D.E. (2009). SCA: A Semantic Conflict Analyzer for Parallel Changes. In Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The foundations of Software Engineering (ESEC/FSE '09). ACM, New York, NY, USA, 291-292.

[25] Fontana, F.A., and Zanoni, M. (2014). Tracking Line Changes in Source Code Repositories. In Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '14). ACM, New York, NY, USA, Article 68, 1 pages.

[26] Li, Y., Wang, L., Li, X., and Cai, Y. (2012). Detecting Source Code Changes to Maintain the Consistence of Behavioral Model. In Proceedings of the 4th Asia-Pacific Symposium on Internetware (Internetware '12). ACM, New York, NY, USA, Article 7, 6 pages.

[27] Omori, T., and Maruyama, K. (2008). A Change-Aware Development Environment by Recording Editing Operations of Source Code. In Proceedings of the 2008 International Working Conference on Mining Software Repositories (MSR '08). ACM, New York, NY, USA, 31-34.

[28] Padhye, R., Mani, S., and Sinha, V.S. (2014). NeedFeed: Taming Change Notifications by Modeling Code Relevance. In Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (ASE '14). ACM, New York, NY, USA, 665-676.

[29] Parnin, C., and Gorg, C. (2008). Improving Change Descriptions with Change Contexts. In Proceedings of the 2008 International Working Conference on Mining Software Repositories (MSR '08). ACM, New York, NY, USA, 51-60.

[30] Sarma, S., Branchaud, J., Dwyer, M.B., Person, S., and Rungta, N. (2014). Development Context Driven Change Awareness and Analysis Framework. In Companion Proceedings of the 36th International Conference on Software Engineering (ICSE Companion 2014). ACM, New York, NY, USA, 404-407.

[31] Servant, F., and Jones, J.A. (2012). History Slicing: Assisting Code-Evolution Tasks. In Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering (FSE '12). ACM, New York, NY, USA, Article 43, 11 pages.

[32] Braun, B. (2008). SAVE: Static Analysis on Versioning Entities. In Proceedings of the Fourth International Workshop on Software Engineering for Secure Systems (SESS '08). ACM, New York, NY, USA, 25-32.

[33] Erdweg, S., Lichter, M., and Weiel, M. (2015). A Sound and Optimal Incremental Build System with Dynamic Dependencies. SIGPLAN Not. 50, 10 (October 2015), 89-106.

[34] Anastasopoulos, M. (2009). Increasing Efficiency and Effectiveness of Software Product Line Evolution: An Infrastructure on top of Configuration Management. In Proceedings of the Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops (IWPSE-Evol '09). ACM, New York, NY, USA, 47-56.

[35] Buchmann, T., Dotor, A., and Westfechtel, B. (2013). MOD2-SCM: A Model-Driven Product Line for Software Configuration Management Systems. Inf. Softw. Technol. 55, 3 (March 2013), 630-650.

[36] Kaur, P., and Singh, H. (2009). Version Management and Composition of Software Components in Different Phases of Software Development Life Cycle. SIGSOFT Softw. Eng. Notes 34, 4 (July 2009), 1-9.

[37] Ki, Y., and Song, M. (2009). An Open Source-Based Approach to Software Development Infrastructures. In Proceedings of the 2009 IEEE/ACM International Conference on Automated Software Engineering (ASE '09). IEEE Computer Society, Washington, DC, USA, 525-529.

[38] Kogel, M. (2008). Towards Software Configuration Management for Unified Models. In Proceedings of the 2008 International Workshop on Comparison and Versioning of Software Models (CVSM '08). ACM, New York, NY, USA, 19-24.

[39] Nguyen, T.T., Nguyen, H.A., Pham, N.H., Al-Kofahi, J.M.,and Nguyen, T.N. (2009). Clone-Aware Configuration Management. In Proceedings of the 2009 IEEE/ACM International Conference on Automated

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Software Engineering (ASE '09). IEEE Computer Society, Washington, DC, USA, 123-134.

[40] Mikkonen T., and Nieminen, A. (2012). Elements for a Cloud-Based Development Environment: Online Collaboration, Revision Control, and Continuous Integration. In Proceedings of the WICSA/ECSA 2012 Companion Volume (WICSA/ECSA '12). ACM, New York, NY, USA, 14-20.

[41] IEEE 828 - IEEE Standard for Configuration Management in Systems and Software Engineering. (2012). The Institute of Electrical and Electronics Engineers. (71 pages).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Appendix 1: Questionnaire On Software Configuration Management Approaches In Malaysia

QUESTIONNAIRE ON SOFTWARE CONFIGURATION MANAGEMENT APPROACHES IN MALAYSIA

Dear Sir/ Madam,
We are investigating the approaches used by the Malaysian government agencies, Institutes of Higher Learning and IT companies in managing software projects. Specifically, we are interested to identify the processes and methods used to manage changes to software product at your organization. We would be grateful if you could spend a few minutes of your time to answer this questionnaire. All information is regarded as *confidential*, and no specific reference to the respondent's e-mail, designation or organization will be made. Please do not hesitate to contact us if you need more information regarding this questionnaire. Your time and participation is much appreciated. Thank you.

SECTION 1: ORGANIZATION BACKGROUND

1. What is the type of your organization?
   - ☐ Government (Ministry/ Department/ Agency/ etc.)
   - ☐ Institute of Higher Learning (IPTA/ IPTS)
   - ☐ Public Limited Company (Berhad)
   - ☐ Private Limited Company (Sdn Bhd)
   - ☐ Sole Proprietorship (Enterprise)
   - ☐ Foreign Company
   - ☐ Others (Please Specify)

2. What is the nature of your business?
   (Select all that apply)
   - ☐ Communication & Networking
   - ☐ Consultancy & Professional Services
   - ☐ Creative Design/ Content
   - ☐ Distributor & Retailer
   - ☐ Education & Training
   - ☐ Internet Based Business
   - ☐ IT Outsourcing
   - ☐ Maintenance
   - ☐ Mobile & Wireless
   - ☐ Network Security
   - ☐ Software Development/ Systems Integrator
   - ☐ Others (Please Specify)

3. What is the total number of full-time employees in your organization?
   - ☐ More than 75
   - ☐ Between 30 to 75
   - ☐ Less than 30

4. Where is your organization located?
   (Kindly indicate location of the main office if your organization has more than one branch)
   - ☐ FT Kuala Lumpur
   - ☐ FT Labuan
   - ☐ FT Putrajaya
   - ☐ Johor
   - ☐ Kedah
   - ☐ Kelantan
   - ☐ Melaka
   - ☐ Negeri Sembilan
   - ☐ Pahang
   - ☐ Penang
   - ☐ Perak
   - ☐ Perlis
   - ☐ Sabah
   - ☐ Sarawak
   - ☐ Selangor
   - ☐ Terengganu

5. How many software projects have your organization involved with, in the last 12 months? *

SECTION 2: CONFORMANCE

6. What type of conformance issues that are usually imposed by the **EXTERNAL STAKEHOLDERS** in software projects?

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

(External stakeholders are individuals, groups or organisations that have an interest and affected by the software product such as the customers, investors, government, etc.)

- ☐ Conformance to specific standards/policies/procedures/etc.
- ☐ Resources (hardware/software/technology/tools/training/etc.).
- ☐ Schedules (development time/project milestones/etc.)
- ☐ Development strategies (versioning/baseline releasing/etc.).
- ☐ Software configuration management.
- ☐ Quality assurance.
- ☐ Others (Please specify):

7. What type of conformance issues that are usually imposed by the **INTERNAL STAKEHOLDERS** in software projects?
(Internal stakeholders are individuals, groups or departments in the organization that have an interest and affected by the software product such as the owners, management, employees, etc.)
   - ☐ Conformance to specific standards/policies/procedures/etc.
   - ☐ Resources (hardware/software/technology/tools/training/etc.).
   - ☐ Schedules (development time/project milestones/etc.)
   - ☐ Development strategies (versioning/baseline releasing/etc.).
   - ☐ Software configuration management.
   - ☐ Quality assurance.
   - ☐ Others (Please specify):

8. Are these constraints (external and internal) documented?
   - ☐ Yes
   - ☐ No

9. Are **SOFTWARE QUALITY FACTORS** explicitly identified in software projects?
(Software quality factors are characteristics of a software product that provide business value such as usability, reliability, security, etc.)
   - ☐ Yes
   - ☐ No

   If yes, kindly state the SOFTWARE QUALITY FACTORS that are usually identified:

10. How frequent are the following SOFTWARE QUALITY FACTORS used in software projects?

	Never	Rarely	Occasionally	Frequently	Always
Functional Suitability *(Degree to which a software product provides functions that meet stated needs)*					
Performance Efficiency (Performance of a software product relative to the amount of resources used)					
Compatibility (Degree to which a software product can exchange information with other products and perform its functions, while sharing the same hardware or software environment)					
Usability (Degree to which a software product achieves specified needs with effectiveness/efficiency/satisfaction)					
Reliability (Degree to which a software product performs specified functions under specified conditions for a specified period of time)					
Security (Degree to which a software product protects information so that persons/other products can only access data based on levels of authorization)					
Maintainability (Degree of effectiveness and efficiency in which a					

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

software product can be modified/improved/corrected/adapted to changes in the environment and requirements)					
Portability (Degree of effectiveness and efficiency in which a software product can be transferred from one hardware/software/ environment to another)					

11. Are the SOFTWARE QUALITY FACTORS in software projects documented?
    - Yes
    - No

12. Are the techniques for assessing SOFTWARE QUALITY FACTORS documented?
    - Yes
    - No

SECTION 3: CONTROL

13. What type of tool is usually used in software projects at your organization?
    - Audit Tools
    - Project Monitoring Tools
    - Project Management Tools
    - Project Reporting Tools
    - Software Configuration Management Tools
    - Software Development Tools
    - Quality Assessment Tools
    - Others (Please specify):

14. Are the use of these tools documented? *
    - Yes
    - No

15. Does your organization control changes to software **ARTIFACTS**?
    (Artifacts are tangible by-products, created during the software life-cycle such as requirements, designs, source code, etc.)
    - Yes
    - No

    If you answer **Yes**, kindly indicate the type of ARTIFACTS that are usually controlled:
    - Requirement Specification Documents
    - Business/Process Logic
    - Design Documents (drawings/ interface descriptions/ process descriptions/etc.)
    - Product Documentation
    - Source Code
    - Audits (plans/ procedures/results)
    - Tools
    - Others (Please specify):

16. Are these (controlled) ARTIFACTS documented?
    - Yes
    - No

17. Who determines the type of ARTIFACTS to be controlled?
    - External Stakeholders
    - Internal Stakeholders
    - Others (Please specify):

3/6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

18. Does your organization have a specific policy or procedure for **CHANGE REQUESTS**?
(A change request is a request to expand/reduce the project scope, processes, procedures, etc., during the product life-cycle).
- ☐ Yes
- ☐ No

19. How does a CHANGE REQUEST communicated in your organization?
- ☐ Verbally
- ☐ In Writing (letter/memo/e-mail/etc.)
- ☐ Using standardized documents (Software Trouble Report/Software Change Request Form/ etc.)
- ☐ Others (Please specify):

20. Who processes the CHANGE REQUESTS (accepting/rejecting)?
- ☐ A Formal Committee
- ☐ The Project Manager
- ☐ Other Internal Stakeholders
- ☐ Other External Stakeholders
- ☐ Others (Please specify):

21. Where do the majority of CHANGE REQUESTS come from?
- ☐ External Stakeholders
- ☐ Internal Stakeholders
- ☐ Others (Please specify):

SECTION 4: AUDIT

22. What type of **AUDIT** is usually carried out in a software project?
(Audit is an examination of a software product or software process to assess compliance with specifications, standards, and other criteria).
- ☐ Formal Audits (Functional Configuration Audit/Physical Configuration Audit/etc.)
- ☐ Informal Audits (in-process audit/etc.)
- ☐ Others (Please specify):

23. How frequent are AUDITS carried out for a particular software project? *
- ☐ Once
- ☐ Daily
- ☐ Weekly
- ☐ Monthly
- ☐ Periodically
- ☐ As needed
- ☐ Others (Please specify):

24. Who determines the type of AUDIT and when it is administered?
- ☐ External Stakeholders
- ☐ Internal Stakeholders
- ☐ Others (Please specify):

25. Does your organization carry out a separate AUDIT for assessing the quality of the software product?
- ☐ Yes
- ☐ No

4/6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

If your answer is YES, please indicate the techniques that are used to assess the quality of the software product?

- ☐ Black-Box Testing
- ☐ Browser Compatibility Testing
- ☐ Configuration Testing
- ☐ Discovery
- ☐ Expert Review
- ☐ Feature Testing
- ☐ Inspection
- ☐ Load Testing
- ☐ Mobile Device Testing
- ☐ Others (Please Specify):
- ☐ Network Testing
- ☐ Operating System Testing
- ☐ Penetration Testing
- ☐ Remote Testing
- ☐ Requirement-Based Testing
- ☐ Security Assessment
- ☐ Stress Testing
- ☐ Use-Cases
- ☐ Vulnerability Scan
- ☐ Walkthrough

26. Are the results of AUDITS documented?
    - ☐ Yes
    - ☐ No

SECTION 5: DELIVERY

27. Does your organization use a software library or central repository in software projects?
    - ☐ Yes
    - ☐ No

28. Can your organization reproduce previous versions of a software product with ease?
    - ☐ Yes
    - ☐ No

29. What type of ARTIFACTS that is usually included in **SOFTWARE DELIVERY**?
    (Software delivery is the identification, packaging, and delivery of artifacts related to a software product such as executable program, product documentation, release notes, etc.)
    - ☐ Artifact Documentation
    - ☐ Change Request Documentation
    - ☐ Conformance Documentation
    - ☐ Executable Program
    - ☐ Product Documentation
    - ☐ Release Notes
    - ☐ Software Configuration Management Documentation
    - ☐ Software Quality Documentation
    - ☐ Others (Please specify):

30. How many software projects could not be delivered on time or required more time to be completed in the last 12 months?

31. How many software projects were revised due to software quality issues in the last 12 months?

32. In the event of a software project could not be delivered on time, how long, on average (monthly), would it take for it to be completed?

SECTION 6: SOFTWARE CONFIGURATION MANAGEMENT

33. Does your organization practise the **VERSIONING** of artifacts?
    (A version is a state of an artifact, snapshot during a specific time in the software project)
    - ☐ Yes
    - ☐ No

5/6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

34. Does your organization releases **BASELINES** in software projects?
(Baseline is a formally approved versions of artifacts such as 'functional baseline' and 'product baseline')
- ☐ Yes
- ☐ No

35. What type of report that is usually generated and used throughout a software project?
- ☐ Artifacts Report/Documentation
- ☐ Audit Report/Documentation
- ☐ Change Request Report/Documentation
- ☐ Conformance Report/Documentation
- ☐ Software Development Report/Documentation
- ☐ Software Quality Assurance Report/Documentation
- ☐ Tools Report/Documentation
- ☐ Others (Please specify):

36. Does your organization use a specific tool for reporting purposes?
- ☐ Yes
- ☐ No

If your answer is YES, kindly list the tool(s) that is usually used:

37. Are you familiar with the term **SOFTWARE CONFIGURATION MANAGEMENT**?
(Software Configuration Management is used to track and control changes to software products during the software life-cycle)
- ☐ Yes
- ☐ No

38. Does your company adopt any specific SOFTWARE CONFIGURATION MANAGEMENT approaches or tools?
- ☐ Yes
- ☐ No

If your answer is YES, kindly specify the approaches/ tools used:

SOFTWARE CONFIGURATION MANAGEMENT ensures that the current design and build state of a software is known. Accurate historical records are useful for the managing development activities and software audit. One of the strength of SOFTWARE CONFIGURATION MANAGEMENT lies in the ability to correctly rebuild a software system in the event of failure.

39. Do you think that SOFTWARE CONFIGURATION MANAGEMENT could improve the overall process and software quality in software projects at your organization?
- ☐ Yes
- ☐ No

If your answer is NO, kindly elaborate:

40. Do you think that SOFTWARE CONFIGURATION MANAGEMENT could improve the management of software projects in your organization?
- ☐ Yes
- ☐ No

If your answer is NO, kindly elaborate:

END OF QUESTIONNAIRE
Thank you for participating

6/6