Integrated Software Management and Product Engineering: The Nigerian Case Study

Aregbesola, Moses Kehinde

Department of Mathematics and Computer Science, Elizade University, Ilara-Mokin, Ondo State, Nigeria

kehinde.aregbesola@elizadeuniversity.edu.ng

Abstract

Integrated Software Management (ISM) and Software Product Engineering (SPE) are the two key process areas (KPA) of interest in the current study. They are both at the defined level of software process maturity. Following the clamor for software companies with high levels of maturity, the study examined the level of performance and relationship between ISM and SPE practices in the software industry of a typical developing country, with Nigeria as the case study. The study was conducted using survey research, case study and action research methodologies. The study covered 30 software companies spanning across several different geographical zones. The results of the research revealed a strong relationship between the ISM and SPE KPAs and a poor performance of the key practices associated with the respect KPAs in the industry. To improve the current level of performance of the respective KPAs, software companies have been advised to procure the services of professional ISM and SPE service providers and to properly equip their software product engineers and integrated software managers with the specialized skills and knowledge necessary for properly discharging their duties.

Keywords: Capability Maturity Model, CMMI, Maturity Level, Software Process, Software Process Engineering, Software Product Engineering, SPE, Integrated Software Management, ISM, Software Industry, Nigeria.

1. Introduction

Both Integrated Software Management (ISM) and Software Product Engineering (SPE) are two key process areas at the Software Engineering Institute's (SEI) Capability Maturity Model Integration (CMMI) maturity 1evel 3 (defined). While ISM is in the management category, SPE is in the engineering category. Glover and Dennie (2017) and O'Neill (2017) described the CMMI as an outstanding performance improvement framework for practical organizations that desire to achieve high performance in their processes. The CMMI is made up of 5 maturity levels. Levels 1 through 5 are respectively named Initial, Managed, Defined, Quantitatively Managed, and Optimizing. These maturity levels each consist of Key Process Areas (KPA), each of which in turn consists of key practices (CMMI Product Team, 2006; Glover and Dennie, 2017; Hurst, 2017; Institute of Configuration Management, 2003).

Integrated Software Management (ISM) has the aim of integrating the software engineering and management activities into an articulate, clearly well-defined software process that is personalized (for the specific project) from the organization's standard software process and associated process resources that are defined in the organization process definition (OPD) KPA (Paulk *et al.*, 1993, Aregbesola, 2017c). How the activities of a project's defined software process will be implemented and managed is usually determined by the software development plan. The software development plan in turn is usually based on the project's defined software process. Also, the tasks of the projects defined software process are closely knit with the management of the software project's resources, cost, schedule, size, effort, and staffing. Lessons-learnt and associated data from the organizational standard software process can be effectively employed in the projects defined software process since latter is crafted out of the former (Paulk *et al.*, 1993; 1995; Russwurm and Meyer, 2000).

The interdisciplinary aspect of ISM that extends beyond software engineering is known as Intergroup Coordination (Aregbesola, 2017d). It emphasizes that not only is the software process integrated, but that the software engineering group's interactions with other groups must be controlled and synchronized (Paulk *et al.*, 1993). ISM on the other hand emphasizes the anticipation of problems and acting to avoid such problems, or mitigate their attendant effects. The practices of the ISM key process area are dependent on the practices associated with planning, tracking, and estimating software projects. The latter practices are explained in the Software Project Planning and Software Project Tracking and Oversight key process areas (Aregbesola, 2017a; Paulk *et al.*, 1995). These key process areas lay emphasis on identifying problems when they happen and making the necessary adjustments to the plans and actions required to address the problems. ISM however ensures that such problems are identified and avoid or mitigate (Paulk *et al.*, 1993; 1995; Clark, 1997).

Since the major goal for most organizations is to attain a Level 3 maturity (Royce, 2002), a number of companies have evolved with the sole aim of helping organisation achieve this maturity level. The ISM service provider companies have developed a wide variety of automated ISM systems and are usually in stiff competition. Some of the desirable features of ISM service provision in the industry has been highlighted by Cortech Developments (nd) to include the following: Self check; Self-financing; Security and stability; Requiring minimum operator training; Cost and time effectiveness; Free product enhancements as they become available; Enforcement of user accountability; Provision of reporting features; Support for State-of-the-art technologies; and Regular upgrades.

Computer Associates, CA Technologies (2013), developed an automated ISM solution that directly addresses the requirements of a large proportion of their clients. The ISM system which was named CA Endevor SCM covered all functional areas of software management including change identification, change management and release management. The system presented its users with a comprehensive framework within which every change activity was automatically supervised and coordinated, starting from specification through production, spanning the complete software development life cycle. Some of the benefits of automated software management over the manual approach include accuracy, reliability, speed, manageability, cost-effectiveness and time management. It resolves the monotony commonly encountered in performing repetitive and time-consuming chores such as comparing software versions from release to release, detecting problems as they occur and tracking change information. Thus, allowing skilled personnel to devote their time more productively by engaging in strategic planning impact analysis, and project management (CA Technologies, 2013).

Software Product Engineering (SPE) is aimed at steadily performing a clearly defined engineering process that integrates the entire software engineering activities to effectively and efficiently bring about consistently correct software products. Technical software project development activities such as requirements analysis, design, coding, and testing are described by SPE (Paulk *et al.*, 1993; Clark, 1997). The analysis of the system requirements allocated to software forms part of software engineering tasks. The systems requirements in question are explained in the Requirements Management key process area (Aregbesola, 2017b). This process includes software requirements development, software architecture development, software design, coding, software integration, and testing. Unit or component testing, integration testing, and systems testing are some approaches employed to ensure that the software meets the specified systems and user requirements (Paulk *et al.*, 1993; 1995; Russwurm and Meyer, 2000).

In the industry, Software Product Engineers are typically saddled with the responsibility of performing the SPE tasks associated with its key practices. Performing these tasks require some specialized skills. Tockey (1998) outlined some of these requisite skills and knowledge to include: Task kick-offs, previews or readiness reviews; Process definition and process improvement techniques; Peer reviews, inspections or walkthroughs; Software testing techniques; Requirements tracing or quality function deployment; Technology innovation; Statistical process control; Proofs of correctness; and Software project audits. Companies such as TATA Consultancy Services (2017) and AgreeYa (nd) have software product engineering services as their sole business focus. They cover the complete range of product engineering, including development, testing and assurance, maintenance and end-of-life support. These companies often provide services that address solution development, product and platform engineering, product and platform development, product portfolio management, product testing and quality assurance, end-of-life support, integration support and innovative next generation product engineering support. AgreeYa (nd) itemized the benefits of their software product engineering services to include: Quick time-to-market; Incremental development and delivery model which involves customers' participation throughout the project's lifecycle; Reduced overall cost of development; employment of the most appropriate technologies and frameworks; and Capability for improving products by incorporating feedback from the market place.

Bavani (2011) explained that numerous challenges (the Y2K issues and the dot-com bubble) faced the software industry at the buildup to the current millennium which significantly challenged the SPE community. These challenges led to the folding up of many startup companies due to shortage of extra funds. Despite these challenges, some of the most significant influences that positively affected SPE during the last decade include: Open source movement; Growth of e-commerce as well as online systems; Agile software development; Collaboration tools; Global software development; Service orientated business models; Modern software engineering; Test automation; and Business intelligence.

The following quotations were lifted from the work of Paulk *et al.* (1993):

The purpose of Integrated Software Management is to integrate the software engineering and management activities into a coherent, defined software process that is tailored from the organization's standard software process and related process assets … this tailoring is based on the business environment and technical needs of the project, as described in Software Product Engineering …

Software Product Engineering involves performing the engineering tasks to build and maintain the software using the project's defined software process (which is described in the Integrated Software Management key process area) and appropriate methods and tools.

The statements explicitly show a strong relationship between Integrated Software Management and Software Product Engineering. The current study is therefore focused on exploring this relation between ISM and SPE, using the Nigeria software industry as a case study. The relationship was explored by means of a study to see the level at which the key practices within the two KPAs are performed in the industry and to know whether or not the performance or non-performance of one of the KPAs affected the other. The approach is expatiated upon in the subsequent sections of this work. This work therefore joins the league of other studies such as Aregbesola and Akinkunmi (2010a; 2010b), Aregbesola *et al.* (2011), Aregbesola and Onwudebelu (2011), and Aregbesola and Oluwade (2014) in exploring the software practices in the Nigerian software industry.

2. Research Methodology

An abridged version of the verified SEI Maturity Questionnaire (Zubrow *et al.*, 1994) was employed as the research tool for gathering necessary data for the study. The questionnaire was administered to software engineers and was completed based on the level of implementation of the key practices within the Integrated Software Management (ISM) and Software Product Engineering (SPE) key process areas. The questionnaire was made up of two major sections. The first sections consisted of questions regarding software process key practices within the individual organisation. While the second section, which was the response section, consisted of four response options: "Yes", "No", "NA" for Not Applicable and "DK" for Don't Know. These four were the response options available to each respondent with regards to the organizations performance of the respective key practices in the initial questions section.

The study was conducted across 30 different software companies, and companies with software arms from different states in Nigeria. Results from only 26 of the companies were eventually used in the final study due to response rate. Some of the selected companies for the research were chosen for further study using the case study and action research approach to further gather and clarifying details for elicited information.

3. Results of the Study

The results of the current study are as shown in Tables 1 and 2. The results are equally graphically represented as depicted by Figures 1 and 2. The results are presented in percentages of actual responses. The averages for each response option are shown in bold at the last row of each table. Discussions and resultant conclusions from these results are presented in the subsequent sections.

Table 1: Software Product Engineering (SPE) Key Process Area

Questions (Key Practices)	**Responses**			
	Yes %	**No %**	**NA %**	**DK %**
I. Are the software work products produced according to the project's defined software process?	23	54	15	8
II. Is consistency maintained across software work products (e.g., is the documentation tracing allocated requirements through software requirements, design, code, and test cases maintained)?	8	77	15	0
III. Does the project follow a written organizational policy for performing the software engineering activities (e.g., a policy which requires the use of appropriate methods and tools for building and maintaining software products)?	4	69	15	12
IV. Are adequate resources provided for performing the software engineering tasks (e.g., funding, skilled individuals, and appropriate tools)?	23	54	12	12
V. Are measurements used to determine the functionality and quality of the	15	65	8	12

software products (e.g., numbers, types, and severity of defects identified)?				
VI. Are the activities and work products for engineering software subjected to SQA reviews and audits (e.g., is required testing performed, are allocated requirements traced through the software requirements, design, code and test cases)?	8	73	4	15
Average	**13**	**65**	**12**	**10**

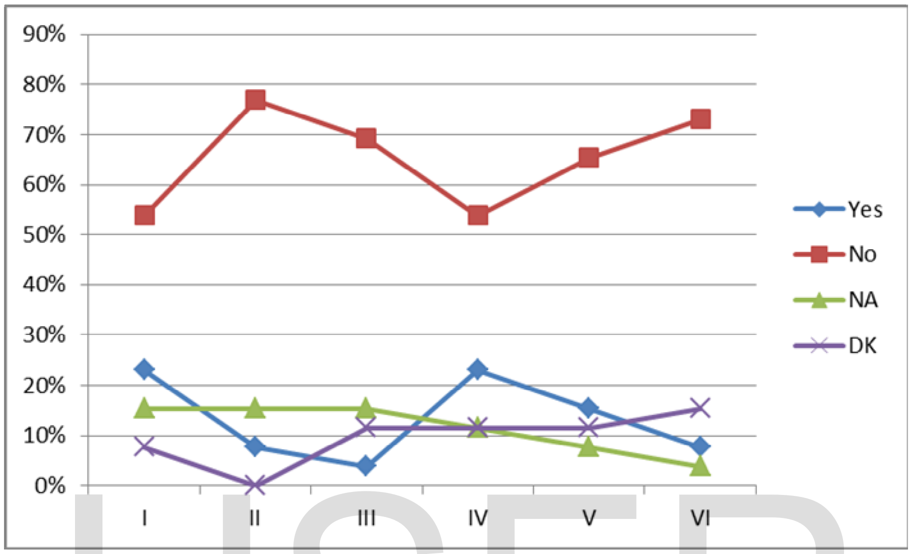

Figure 1: Chart of Key Practices for Software Product Engineering

Table 2: Integrated Software Management (ISM) Key Process Area

Questions (Key Practices)	**Responses**			
	Yes %	**No %**	**NA %**	**DK %**
I. Was the project's defined software process developed by tailoring the organization's standard software process?	4	62	19	15
II. Is the project planned and managed in accordance with the project's defined software process?	8	27	38	27
III. Does the project follow a written organizational policy requiring that the software project be planned and managed using the organization's standard software process?	0	92	8	0
IV. Is training required for individuals tasked to tailor the organization's standard software process to define a software process for a new project?	12	27	54	8
V. Are measurements used to determine the effectiveness of the integrated software management activities (e.g., frequency, causes and magnitude of replanning efforts)?	12	58	12	19
VI. Are the activities and work products used to manage the software project subjected to SQA review and audit?	0	73	19	8
Average	**6**	**56**	**25**	**13**

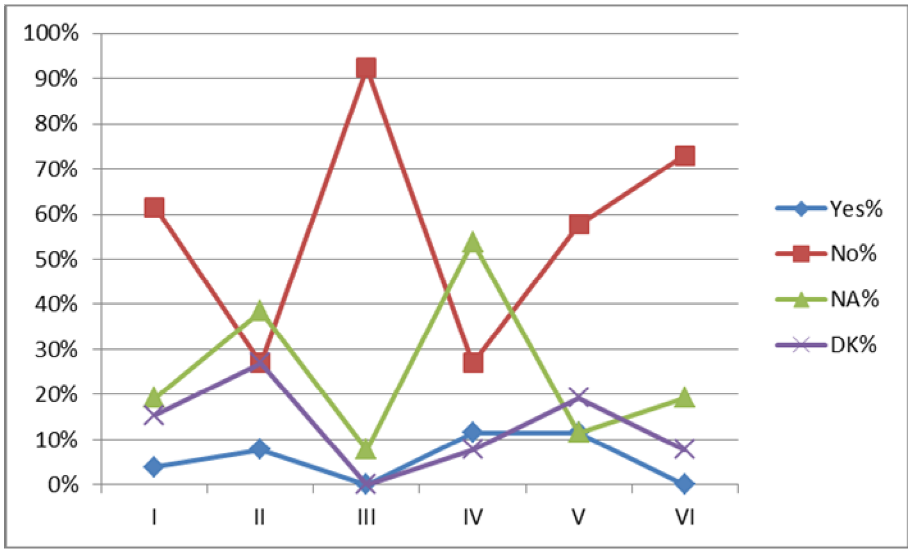

Figure 2: Chart of Key Practices for Integrated Software Management

4. Discussion of Results

The results depicted in Tables 1 and 2 as well as in the corresponding charts of Figures 1 and 2, show a high degree of non-performance of key practices in the Software Product Engineering (SPE) and Integrated Software Management (ISM) key process areas. SPE recorded a meager 13% performance and a whopping 65% non-performance, while ISM recorded 6% and 56% for performance and non-performance respectively. The poor performance of these KPAs might not be unconnected with the postulation of Bavani (2011), that numerous challenges in the software industry following the buildup to the current millennium significantly challenged software product engineering community. Also, these KPAs are associated with the software process maturity at level 3 (Defined) and could therefore account for their relatively low implementation since the Nigeria software industry is currently considered to be at maturity level 1 according to the study of Aregbesola and Akinkunmi (2010a; 2010b), Aregbesola *et al.* (2011), Aregbesola and Onwudebelu (2011), and Aregbesola and Oluwade (2014).

Since ISM has the aim of integrating the software engineering and management activities into an articulate, clearly well-defined software process that is personalized, for the specific project, from the organization's standard software process, the low performance of ISM is an indication of a lack of synchronization across the different software engineering and management units. Except of course, if the entire project team is very small, perhaps too small a team (with several overlapping roles) to be divided into the traditional units, as is mostly the case in many Nigerian software companies (Soriyan and Heeks, 2004).

In a similar light, since Software Product Engineering (SPE) is aimed at steadily performing a clearly defined engineering process that integrates the entire software engineering activities to effectively and efficiently bring about consistently correct software products, the poor performance of SPE practices could imply a number of things. The first is that the software engineering activities are not properly integrated, or as in the case of ISM, it is possible that the entire project team is very small, perhaps too small a team (with several overlapping roles) to be divided into the traditional units, hence having minimal requirement for integration. Secondly, the poor performance of the SPE KPA could also imply that the software products are not being developed in an effective or efficient manner, or that the resulting products are not consistently correct. Whatever the case, not only is the poor performance of SPE detrimental to the final software products, but also the software companies and their clients.

The results of the current study were obtained by exploring the relationship between ISM and SPE KPAs in the Nigerian software industry. The relationship was explored by observing the level at which the key practices within the two KPAs were performed in the industry and to deduce whether or not the performance or non-performance of one of the KPAs affected the other. The results of the study have affirmed the existence of a strong relationship between integrated software management and software product engineering since the observed performance trends for both KPAs are similar, with one affecting the other, and with equally similar possible causes.

5. Conclusion

The presented study focused entirely on the exploration of the performance and relationship between two KPAs at the defined level of software process maturity, namely, integrated software management and software product engineering. Survey, case study and action research were the methodologies employed in carrying out the research. The study has affirmed that the two KPAs are dependent upon each other. Hence, the performance or nonperformance of one affects the other.

The study equally showed that the level of performance of the aforementioned KPAs in the Nigerian software industry is quite weak. These KPAs should therefore be accorded the needed attention since the major goal for most organizations is to attain a Level 3 maturity (Royce, 2002). It is advisable that the software companies employ the services of ISM and SPE servicing companies, such as CA Technologies (2013) and TATA Consultancy Services (2017), to deploy their services and automated systems for productivity enhancements. This would go a long way in improving the overall maturity level of the industry.

Also, to improve the performance of the respective KPAs, software companies are advised to equip their members of staff, especially software product engineers and integrated software managers with specialized skills and knowledge, such as those outlined by Tockey (1998), required for carrying out the responsibilities they are saddled with.

References

AgreeYa (nd). Software Product Engineering: Consulting, Technology , outsourcing. A CMMI Level 5 Company. Retrieved 18/04/2017 from www.agreeya.com

Aregbesola M. K. (2017a). Software Project Planning with Tracking and Oversight. Manuscript submitted for publication.

Aregbesola M. K. (2017b). Exploring Requirements Management with Software Subcontract and Configuration Management. Journal submitted for publication.

Aregbesola M. K. (2017c). Experiential Appraisal of Organizational Process Focus and Process Definition in Nigerian Software Companies. Journal of Scientific and Engineering Research. In press.

Aregbesola M. K. (2017d). Investigating Training Program and Intergroup Coordination in relation to Peer Review in Nigerian Software Companies. Journal of Scientific and Engineering Research. In press.

Aregbesola M. K. and Akinkunmi B. O. (2010a). Software Process Implementation – A focus on the Nigerian Software Industry. Journal of Research in Physical Sciences, Vol. 6, No. 2, pp. 9 – 14.

Aregbesola M. K. and Akinkunmi B. O. (2010b). Software Process Implementation – A focus on the Nigerian Software Industry. International Research and Development Institute (IRDI), World Congress on Research and Development, Conference Center, University of Ibadan, 5th - 8th October. Vol. 5, No. 6, pg.111-116.

Aregbesola M. K. and Oluwade B. A. (2014). An Experimental Evaluation of Defect Prevention and Change Management in Software Process Optimization in the Nigerian Software Industry. ARPN Journal of Systems and Software Vol.4, No.1, pp. 5-11.

Aregbesola M. K. and Onwudebelu U. (2011). Typical Software Quality Assurance and

Quality Management Issues in the Nigerian Software Industry. National Association for Science, Humanities & Education Research, 8th National Conference, University of Ado Ekiti, Ado Ekiti, September 14-17.

Aregbesola M. K., Akinkunmi B. O., and Akinola O. S. (2011). Process Maturity Assessment of the Nigerian Software Industry. International Journal of Advances in Engineering and Technology (IJAET), Vol.1, Issue 4, pp. 10-25.

Bavani R. (2011). The 10 best influences on software product engineering. Columns SD Times. February. Retrieved 18/04/2017 from www.sdtimes.com.

CA Technologies (2013). CA Endevor®/DB for CA IDMS™.Concepts and Facilities Guide Release 18.5.00. Retrieved 18/04/2017 from http://ca.com/

Clark, K. B. (1997). The Effects of Software Process Maturity on Software Development Effort. A PhD (Computer Science) Dissertation Presented to the Faculty of the graduate school, University of Southern California.

CMMI Product Team (2006). CMMI for Development, Version 1.2 - CMMI-DEV, V1.2. Software Engineering Institute, Carnegie Mellon University.

Cortech Developments (nd). software integration solutions: security, fire and energy management. Retrieved 18/04/2017 from www.cortech.co.uk.

Glover M. T. and Dennie D. (2017). CMMI–Agile Process Combo: How to be Agile with CMMI. Excellence in Measurement Technology.

Hurst J. (2017). The Capability Maturity Model and Its Applications. SANS Software Security with Frank Kim.

Institute of Configuration Management (2003). SEI's Capability Maturity Model Integrated (CMMI) Relative to ICM's CMII (Rev B). White Paper. Retrieved 18/04/2017 from www.icmhq.com.

O'Neill D. (2017). In Search of a Modern Software Life Cycle: Secure DevOps Foundations for Large-Scale Software Systems. CrossTalk March/April 2017: Modern Process Trends.

Paulk M. C., Weber C. V., Curtis B., & Chrissis M. B. (1995). The Capability Maturity Model: Guidelines for Improving the Software Process. Addison – Wesley, Boston.

Paulk M. C., Weber C. V., Garcia S. M., Chrissis M. B., and Bush M. (1993). Key Practices of the Capability Maturity Model, Version 1.1. Technical Report CMU/SEI-93-TR-025 ESC-TR-93-178, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Royce W. (2002). CMM vs. CMMI: From Conventional to Modern Software Management. The Rational E-Zine for the rational community. Retrieved 19/04/2017 from http://www.therationaledge.com/content/feb_02/f_conventionalToModern_wr.html

Russwurm W., and Meyer L. (2000). Integrated Evaluation Procedure for Software/Hardware System Development Processes based on the Software Capability Maturity Model (CMM)‡. Software Process Improvement and Practice; 5: 231–242.

TATA Consultancy Services (2017). Software Product Engineering Services. Communications, Media & Technology:IT Services Business Solutions Consulting. Retrieved 18/04/2017 from www.tcs.com.

Tockey S.(1998). Recommended Skills and Knowledge for Software Engineers. Software Engineering: The Development Process, Vol I, Chapter 1.

Soriyan A. and Heeks R. (2004). A Profile of Nigeria's Software Industry. Development Informatics Working Paper No 21, Institute for Development Policy and Management, University of Manchester, 2004.

Zubrow D., William H., Jane S. and Dennis G. (1994). Maturity Questionnaire. Special Report CMU/SEI-94-SR-7, June 1994.