International Journal of Emerging Trends in Information & Knowledge Management
https://www.eurekajournals.com/knowledge.html
ISSN: 2581-3579
Special Issue: "Transcending Tech Ed Nexus: Advancing Education through Technological Integration" - 12th June-2024

Recognizing Software Engineering Difficulties in Software SMEs

Dr. Jaigopi K

Vijaya College.

Abstract

SSMEs, or small and medium-sized software enterprises, play a critical role in developing economies. They cannot adopt automated software engineering tools or advanced software engineering techniques in the same way as large and ultra-large companies due to their size. Utilising semi-structured interviews with four SSMEs, we investigate the software engineering challenges faced by SME's in Thailand, an emerging software development market. Following a thematic analysis of the interview transcripts, we identified several recurring problems, including inadequate testing, code-related problems, and imprecise effort estimation. We found that SSMEs must embrace modern software engineering best practices like automated testing, continuous integration, and automated code review in order to implement sophisticated automated software engineering tools and procedures.

Keywords: Empirical Study, Case Study, Software SMES.

Introduction

Small and medium-sized businesses, or SMEs, make up 95%-99% of all businesses in 36 OECD member nations and are essential to a nation's economic development [1]. One of the main forces behind the software industry and crucial to a nation's competitiveness and innovation are software SMEs (SSMEs), which includes software startups [2]. For instance, according to a recent study [3], the software market in Thailand is expected to be valued $4 billion in 2020 and has grown steadily over time. Over 8,000 software companies, both large and small, employ over 100,000 software engineers in the nation's software industry [3]. The software sector has emerged as a pillar of the nation's long-term strategy with the implementation, a transformation plan that encourages the growth of technology clusters.

However, SSMEs encounter a number of difficulties when creating their software. According to a technical report [9], configuration management, quality assurance, and project assessment and control are a few of the main areas where SSMEs fall short. SSMEs lack a large workforce, in contrast to large or ultra-large software companies. SSMEs may also continue to employ

expensive and time-consuming manual programming tasks carried out by their programmers, as well as traditional software development methodologies and tools. Therefore, by determining the software engineering difficulties that SMEs encounter, we can suggest cutting-edge automated

SMEs can use software engineering (ASE) approaches to lower costs and accelerate software development and delivery. Advanced software engineering techniques cannot be implemented until the issues causing the flaws are recognised and resolved. In order to close this gap, our research will look into the unique software engineering problems faced by small and medium-sized enterprises (SME) in Asian cultures, particularly in Thailand. Additionally, we will examine the tools that these companies use, with the goal of recommending automated software engineering (ASE) tools and techniques in the future. Owing to the disparate work cultures in the Western region [10], this study offers fresh perspectives on SSMEs in Thailand and potentially Asia.

The Study

This study looks into the methods currently employed in Thai SSMEs' daily software development routines and identifies the time, money, and software quality barriers preventing them from implementing advanced software engineering techniques. Our goal is to examine a small number of entities (SMEs) in their natural environment free from experimental control or manipulation, in accordance with the guidelines established by Benbasat et al. [11]. A case study is therefore a suitable approach for achieving our goal. Utilising semi-structured interviews [12] and thematic analysis [13], we are able to identify the shared difficulties that the businesses face. Identifying research questions is where we begin. After that, we create the interview guide and conduct semi-structured interviews with the staff members of the companies. Finally, we use the transcript to conduct a thematic analysis and identify common software engineering challenges.

A. Research Questions

We aim to answer the following research questions:

1) **RQ1 (SE Challenges):** What difficulties do SSMEs face when developing software on a daily basis? The issues that must be resolved in order to implement advanced software engineering techniques will be identified by the response to this research question.

2) **RQ2 (Present Practices):** Which instruments are in use? We can learn more about the companies' current procedures by taking a look at the tools they are using as a stand-in. We can better align recommended advanced practices, automated tools, and techniques with current practices by knowing the answer to this question.

Table 1: Studied SSMES

Company	Products	No. of Developers
Company A	E-Learning platform	15
ProGaming	Web and mobile games	10
Roots	Enterprise solutions	40
Zwiz.AI	Enterprise AI chatbot	6

B. The Studied Companies

We study the challenges and practices of four Thai SSMEs: ProGaming, Zwiz.AI, Roots, and Company A (pseudonym). Table I provides a summary of the four companies. They offer a variety of software products and services for various business needs, such as chatbots, e-learning, games, and enterprise solutions. An online e-learning platform with over 1,800 courses, 10,000 monthly users, and 280 educators is provided by Company A. Over 50 games created by ProGaming have been downloaded two million times. They also offer Thai organisations game development services. Roots provides business consulting services to a number of major Thai corporations. More than 30,000 companies and more than 10 million users benefit from Zwiz.AI's AI chatbot services. Three of them are very small (less than 25 employees), and all of them are small (less than 50 employees).

C. Interview Design

Examining the difficulties encountered by the four companies' software development teams is the aim of the interviews. Rather than using a predetermined list of questions, we conduct semi-structured interviews that are based on a broad grouping of topics and enquiries [12]. One programmer at a time is spoken with using this method. Every time an interview is conducted, the interview script is revised, particularly if interviewees begin to give the same responses (saturation effect). When no new observations or insights are made, the interview process ends. After recording, the interview is transcribed. Each interview is limited to a maximum of 30 minutes.The executives of the companies recommend the interviewees based on how well-suited they are for the project.

D. Semi-structured Interviews

In February 20201, semi-structured interviews took place at the offices of the companies. The semi-structured interviews were conducted by the first author in a room that was provided by each company during her visit. The recordings of the interviews were transcribed, and the interviews were conducted in Thai. Table II is a list of the opening questions that were asked during the interviews. These were meant to be conversation starters, and the interviewer was free to ask more follow-up questions in response to each interviewee's responses.

Table 2: Initial questions used in the interviews

	Please explain your role at the company.
	How long have you been working at the company?
RQ1	Please explain your day-to-day activity.
	How do you develop the software product?
	What works well in your company’s software development? What would you suggest to improve?
RQ2	Please explain the tools you use during software development.

E. Thematic Analysis

Reflexive thematic analysis, which is frequently used to analyse qualitative data, was applied to the transcript units following the transcription of the interview recordings. We developed the theme and code using a deductive approach, guided by the difficulties encountered in software engineering, as per the recommendations provided by Braun and Clarke [13]. A set of transcripts was given to the first, third, and fourth authors (the coders) to code. After carefully going over each transcript, they gave each transcript unit one or more codes. The codes, which included software engineering difficulties like "lack of unit testing," "coding style violations," and "mismatched requirements," were written in English to aid in generalisation and enable collaboration with authors from various nations.

Results

Twenty software engineers and personnel with a connection to software were interviewed (details are displayed in Table III). The interviewees' roles range from development to management, including Chief Technology Officers (CTOs). The duration of employment varies from four months to eleven years.

A. RQ1: Identified SE Challenges

The identified software engineering challenges are shown in Table III. We discuss each of the challenges in detail below.

TABLE 3: Interviewed participants

Company	Participant (Experience at the company in years) Company A Chief Technology Officer (6), Developer (2), Senior Developer (1), Chief Operating Officer (6), Developer (1.5)
ProGaming	Managing Director & Technical Lead (11), Lead Developer (6), Project Manager (5), Developer (3), Developer (1)
Roots	Chief Technology Officer (5), Senior Developer (4), Senior Developer (2), Developer (2), Developer (0.7)
Zwiz.AI	Chief Technology Officer (2), Developer (2), Developer (1), Developer (0.25), Developer (0.33)

1. *Absence of testing:* During the coding, theme mapping, and interviews, the most noteworthy problem that surfaced was the absence of testing for the software used by the four companies. This comprises the following: inadequate testing (six times cited), inadequate testing (six times cited), absence of automated testing (four times cited in the interviews by different interviewees), and insufficient testing time and expertise (three times cited). Businesses focus mostly on software implementation, with manual testing done at the end of the development process. Not only are unit tests rarely performed, but developers also don't think highly of them when they're working on a project. One interviewee also pointed out the shortcomings of regression testing.
2. *Code-related issues:* Code-related problems are the subject of the second challenge. Each of the four businesses is also faced with this challenge. We discovered the following three

problems: (1) low code quality (mentioned six times), which includes faulty code, duplicate code, difficult-to-understand legacy code, code that is difficult to reuse and requires refactoring, and (3) difficulty maintaining or using coding conventions within the organisation (mentioned four times). The lack of code analysis and measurements is mentioned six times.

3. *Unclear and incomplete requirements:* The third challenge (mentioned 13 times by 3 companies) is about requirements. The interviewees clarified that certain requirements were too ambiguous, imprecise, or lacking. A few of the requirements didn't align with what the customers needed. Additionally, they brought up the problem of "scope creep," which occurs when new requirements are added while a project is being developed.
4. *Inaccurate effort estimation:* The fourth challenge—which has been mentioned eleven times by three companies-is estimating the amount of work that needs to be done. The interviewees clarified a number of points. For instance, they frequently had to work overtime and put in more time than was anticipated. In a sprint, additional unplanned tasks were added. Furthermore, a few interviewees indicated that they were unable to accurately estimate how long the tasks they were given would take to finish.
5. *Lack of knowledge, sharing, and documentation:* A few interviewees (who were mentioned eight times by four different companies) proposed improving documentation and knowledge sharing. For instance, developing strategies for new team members to quickly pick up the necessary knowledge or providing training to improve skills related to the job.
6. *No configuration management:* Software configuration management has certain challenges as well (four mentioned by three different companies). For instance, challenges in handling environment configurations that support various browsers and screen sizes, disparate local development and deployment setups, and a lack of knowledge about stable versions of necessary frameworks or tools.
7. *Bug-inducing changes:* According to three interviewees, new changes brought about more bugs (three companies mentioned this). This is also related to the lack of testing done to find these bugs brought on by the modifications..
8. *Issues during code review:* According to three interviewees, new changes brought about more bugs (three companies mentioned this). This is also related to the lack of testing done to find these bugs brought on by the modifications..
9. *Other challenges:* There are certain challenges that do not fit into the categories listed above. Lack of an automated deployment system, a lack of software processes, and communication problems between teams are a few examples.
10. Overall, it can be said that challenges 2, 5, 7, and 8 are impacted by the absence of testing and the lack of adoption of automated code review. Before more advancements can be made, these modern best practices must be implemented.

B. RQ2: Identified Current Practices

To gain more insights into the current practices of the companies, we asked the participants about automated tools that they regularly use. We observed some interesting findings as follows.

Frameworks for unit testing: The companies did not use unit testing frameworks, even though they employed continuous integration. It's interesting to note that one interviewee mentioned

using Javascript's JEST and Mocha unit testing frameworks. Nonetheless, the absence of unit testing adoption was brought up by the other interviewees from the same organisation. The interviewee may have brought up unit testing frameworks because of their personal experience, which suggests that the organisation still needs to work on improving testing adoption throughout the entire organisation.

Local enforcement applies to formatting and code structure: The interviewees from the four companies clarified that their IDEs included code formatters like Prettier and linters like PyLint or ESLint.

Use of project management tools varies: We saw that the four businesses manage their tasks and monitor the status of their projects using specialised project management platforms like Asana, Trello, or Clickup. Nevertheless, two businesses also mentioned handling some project components with Google Sheets, the all-purpose spreadsheet application.

Mixed use of communication tools: The respondents from three different companies stated that they primarily used the popular Thai chat apps LINE and Discord. Only one business made use of MatterMost, a specialised communication tool. Given that LINE and Discord are primarily personal chat apps, it is possible for the developers to combine work and personal life. This may be partially explained by the Thai culture's tendency for people to react more quickly.

Discussion

Outcomes: The case study is a component of an ongoing, cross-national initiative between industry and academia to assist small and medium-sized enterprises (SMEs) in adopting appropriate automated software engineering tools and techniques, akin to those that are successfully employed in large and ultra-large corporations.

The goal of the study that was presented was to determine the difficulties faced by SMEs in developing nations like Thailand and to use automated software engineering to solve those difficulties. During the course of this project-related study, we identified one specific issue that must be resolved before more sophisticated techniques or automated software engineering can be implemented: a lack of testing. Using automated testing tools to test software in continuous integration environments and automated software engineering are current best practices in software engineering.

Lessons Learned: The observation that SSMEs must first adopt modern software engineering best practices is a crucial lesson learnt before attempting to assist them in implementing automated software engineering tools and techniques. The study's findings have changed the scope and objectives of the ongoing project. Rather than introducing the most cutting-edge automated software engineering tools and techniques, the project will now concentrate on the effects of adopting best practices in software development, such as automated code review, continuous integration, and automated testing.

Takeaway Messages: The success of the adoption of automated software engineering tools and techniques has been largely attributed to knowledge transfer between software engineering

research and industry. Large and ultra-large companies are typically the settings for this kind of cooperation and knowledge transfer. SSMEs frequently lack the capacity to engage in this kind of collaboration and knowledge transfer, particularly in emerging markets. Therefore, in order to prevent the gap between SSMEs and large and ultra-large companies from growing, research should involve SSMEs in order to facilitate the adoption of automated software engineering tools and techniques.

Related Work

The experiences of Laporte et al. [15] and Habra et al. [16] with implementing ISO29110, a set of software process guidelines for very small entities (VSEs), to businesses are reported. The acceptance of software process improvements in VSEs is studied by Basri et al. [17]. Preliminary findings from a gap analysis of 39 Thai VSEs' software development activities are reported by Sunetnunta et al. [9]. They discovered that project evaluation and control, quality assurance, and configuration management are the main areas of weakness. Phongpaibul and Boehm [10] demonstrate how cultural considerations influence Thailand's adoption of software process improvement. According to Tuape and Ayalew [18], technical factors are the main elements influencing African SSMEs' software processes. Additionally, common obstacles to software process improvement in SSMEs are reported by Larrucea et al. [2]. Our research

Threats to Validity

External validity: The study is performed on four Thai SSMEs and the findings may not be generalised to all SSMEs in Thailand and other developing countries. We mitigated this threat by selecting SSMEs from different domains. Due to cultural differences, some of the findings may only be found in Asian regions and not be applied to SSMEs in other regions. **Internal validity:** The selection of participants may affect the validity of the identified challenges. We mitigated this threat by using an inclusion criteria to control the experience with the company’s software development process.

Conclusion

This study examines the practices currently employed by SSMEs and the software engineering challenges they face. Semi-structured interviews and a thematic analysis of four Thai software companies are used in this case study investigation. The outcome demonstrates that the examined companies are encountering difficulties with modern software best practices, such as requirements, code-related problems, and software testing, which are frequently widely used in large or ultra-large software companies. Furthermore, we discovered that rather than utilising specialised tools for project management and communication, SSMEs are making use of all-purpose tools like spreadsheets or instant messaging apps. Software quality is not the only thing that suffers from a lack of testing; advanced practices and automated software engineering tools are also hindered by it.

Acknowledgements

The Newton Fund and the Royal Academy of Engineering, UK (Industry Academia Partnership

Programme - 18/19 - IAPP18-19/74) are the funding sources for this work. The research ethics boards at both universities gave it their blessing. The four companies under study do not hold any financial or proprietary interests for the authors.

References

OECD, *OECD SME and Entrepreneurship Outlook 2005*. OECD Publishing, 2005.

X. Larrucea, R. V. O'Connor, R. Colomo-Palacios, and C. Y. Laporte, "Software Process Improvement in Very Small Organizations," *IEEE Software*, vol. 33, no. 2, pp. 85-89, Mar 2016.

Digital Economy Promotion Agency, "Thailand's Digital Industry Sur- vey 2020 and 3-Year Prediction," https://www.depa.or.th/th/article-view/ index-3-years-since-2563, 2021.

W. D. Royal Thai Embassy, "Thailand 4.0 - Agenda 2: Development of Technology Cluster and Future Industries," https://thaiembdc.org/ thailand-4-0-2, 2016.

J. Wongwuttiwat and A. Lawanna, "The digital Thailand strategy and the ASEAN community," *EJISDC*, vol. 84, no. 3, p. e12024, May 2018.

P. Samuel, "Vietnam's digital transformation plan through 2025," https://www.vietnam-brief ing.com/news/ vietnams-digital-transformation-plan-through-2025.html, 2021.

Ministry of Commerce and Industry (India), "Make In India," https://www.makeinindia.com/, 2014.

K. Crismundo, "Industry 4.0 plans to help PH improve innovation rankings," https://www.pna. gov.ph/articles/1114871, 2020.

T. Sunetnanta, S. Suwannaroj, and P. Sangpar, "ISO/IEC 29110 for Com- petitiveness - Challenges of Digital Cluster Development in Thailand," ISO/IEC JTC 1/SC 7 Software and Systems Engineering Working Group 24, Tech. Rep., August 2016.

M. Phongpaibul and B. Boehm, "Improving quality through software process improvement in Thailand," *ACM SIGSOFT Software Engineering Notes*, vol. 30, no. 4, p. 1, Jul 2005.

I. Benbasat, D. K. Goldstein, and M. Mead, "The case research strategy in studies of information systems," *MIS Quarterly*, vol. 11, p. 369, 1987.

S. B. Merriam and E. J. Tisdell, *Qualitative Research: A Guide to Design and Implementation, 4th Edition*. Jossey-Bass, 2015.

V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77-101, Jan 2006.

A. Rodrigues, A. Bessa, and P. R. Pinheiro, "Barriers to Implement Test Process in Small-Sized Companies," in *Communications in Computer and Information Science*, 2010, vol. 112 CCIS, pp. 233-242.

C. Y. Laporte, R. V. O'Connor, and L. H. G. Paucar, "The Implementa- tion of ISO/IEC 29110 Software Engineering Standards and Guides in Very Small Entities," in *CCIS*, 2016, vol. 599, pp. 162-179.

N. Habra, S. Alexandre, J.-M. Desharnais, C. Y. Laporte, and A. Renault, "Initiating software

process improvement in very small enterprises," *IST*, vol. 50, no. 7-8, pp. 763-771, Jun 2008.

S. B. Basri and R. V. O'Connor, "Organizational commitment towards software process improvement: An Irish software VSEs case study," in *ITSim'10*, vol. 3, 2010, pp. 1456-1461.

M. Tuape and Y. Ayalew, "Factors Affecting Development Process in Small Software Companies," in *SEiA'19*, 2019, pp. 16-23.

M. Felderer and F. Auer, "Software quality assurance during implemen- tation: Results of a survey in software houses from Germany, Austria and Switzerland," *Lecture Notes in Business Information Processing*, vol. 269, pp. 87-102, 2017.

M. Yaseen, S. Baseer, and S. Sherin, "Critical challenges for requirement implementation in context of global software development: A systematic literature review," in *ICOSST'15*, 2016, pp. 120-125.

V. Garousi, M. Felderer, M. Kuhrmann, K. Herkilog˘lu, and S. Eldh, "Exploring the industry's challenges in software testing: An empirical study," *Journal of Software: Evolution and Process*, vol. 32, no. 8, 2020.

C. Ponsard and J.-C. Deprez, "Helping SMEs to better develop soft- ware," in *ICSE'18 (SEIP)*, 2018, pp. 213-214.

S. Jones, J. Noppen, and F. Lettice, "Management challenges for DevOps adoption within UK SMEs," in *QUDOS'16*, 2016, pp. 7-11.

T. Shah and S. V Patel, "A Review of Requirement Engineering Issues and Challenges in Various Software Development Methods," *IJCA*, vol. 99, no. 15, pp. 36-45, 2014.

M. V. Ma¨ntyla¨ and J. Vanhanen, "Software deployment activities and challenges - A case study of four software product companies," in *CSMR'11*, 2011, pp. 131-139.

M. Shahin, M. Zahedi, M. A. Babar, and L. Zhu, *An empirical study of architecting for continuous delivery and deployment*. Empirical Software Engineering, 2019, vol. 24, no. 3.