A comparative study of software development practices in Bangladesh, an emerging country

Partha Chakraborty, Khalid Hasan and Anindya Iqbal

Bangladesh University of Engineering and Technology,
Dhaka – 1000, Bangladesh
Email: shuvopartho@gmail.com
Email: shawontafsir@gmail.com
Email: anindya@cse.buet.ac.bd

Gias Uddin

York University,
Toronto, Ontario, Canada
Email: guddin@yorku.ca

Rifat Shahriyar*

Bangladesh University of Engineering and Technology,
Dhaka – 1000, Bangladesh
Email: rifat@cse.buet.ac.bd
*Corresponding author

Abstract: Software development is complicated with rapidly changing requirements, techniques, processes, and the involvement of diverse stakeholders. A study on the software development practice in a country can reveal the challenges and scopes of improvement for countries of similar types. There are several studies on the software industry in developed countries. No recent study has explored the software development practices and challenges in an emerging country like Bangladesh. We aim to understand the methods and practices adopted and the challenges the software companies face in an emerging country like Bangladesh. We also aim to understand whether and how Bangladesh's development practices and methods differ from other countries. We used insights from semi-structured interviews to design a survey where 137 software practitioners from diverse companies responded. Our findings can guide software practitioners to improve development practices in emerging countries and software providers with new tool support to assist in the process.

Keywords: software development practices and challenges; survey; emerging countries; Bangladesh.

Reference to this paper should be made as follows: Chakraborty, P., Hasan, K., Iqbal, A., Uddin, G. and Shahriyar, R. (2024) 'A comparative study of software development practices in Bangladesh, an emerging country', *Int. J. Software Engineering, Technology and Applications*, Vol. 2, No. 2, pp.149–187.

Biographical notes: Partha Chakraborty is working towards his Doctoral in Computer Science at the University of Waterloo, Canada. His main research interests include empirical software engineering, bug localisation, and software vulnerability detection. He completed his Bachelor of Science (BSc) in November 2018 from the Department of Computer Science and Engineering (CSE) of Bangladesh University of Engineering and Technology (BUET).

Khalid Hasan completed his Bachelor of Science (BSc. in November 2018 from the Department of Computer Science and Engineering (CSE) of Bangladesh University of Engineering and Technology (BUET). He has over four years of professional experience in startups and medium-sized software companies in Dhaka, Bangladesh. Currently, while pursuing an MSc at Missouri State University, he concurrently serves as a graduate assistant, actively engaged in research focused on data mining topics. This journey has equipped him with a solid foundation in software engineering, practical industry insights, and a passion for advancing knowledge through research in software engineering and data mining.

Anindya Iqbal received his BSc and MSc in Computer Science and Engineering (CSE) from the Bangladesh University of Engineering and Technology (BUET), Bangladesh, in 2005 and 2009, respectively. He received his PhD from the Monash University, Australia in 2013. Currently, he is a Professor in the Department of CSE at BUET. His major research interests are software engineering, empirical software engineering, applied machine learning, security and privacy, participatory sensing systems, and wireless sensor networks. He has published over 35 refereed publications in reputed journals and conferences, including *EMSE*, *IST*, *Ad Hoc Networks*, *JNCA*, *ASE*, *ESEM*, *MSR*, *SANER*, etc.

Gias Uddin is a Tenure-Track Assistant Professor at the Department of Electrical Engineering and Computer Science of York University, Canada. He has completed his PhD from the McGill University in 2018 and Master's degree from the Queen's University in 2008. He has been deeply involved with the Canadian software industry and public sector since 2008 in various increasingly senior roles related to software engineering and data science. His research was published in the topmost journals and conferences in SE research (TSE, ICSE, ASE, etc.). His research was covered by BBC News and is currently funded by several external grants (IBM, NSERC, Alberta Innovates, etc.).

Rifat Shahriyar is working as a Professor at the Department of Computer Science and Engineering (CSE) of Bangladesh University of Engineering and Technology (BUET). He completed his PhD in 2015 from the Research School of Computer Science at the Australian National University (ANU). His research interests are memory management (garbage collection), programming language, software engineering, and natural language processing. He has

published over 40 refereed publications in reputed journals and conferences, including *IST*, *EMSE*, *TKDE*, *OOPSLA*, *EMNLP*, *NAACL*, *ICDE*, *CHI*, *ISMM* and *ESEM*.

1 Introduction

A study on software development practices in a country can reveal the challenges and scopes of improvement in software industries of similar countries. There are several studies that focused on understanding the software engineering practices in developed countries with matured software industries (e.g., Canada, Turkey, Netherlands, New Zealand) (Garousi and Zhi, 2013; Garousi et al., 2015; Vonken et al., 2012; Wang and Galster, 2018). However, the software industry of a developed country differs from that of a developing country in many aspects. Easy migration opportunities for software engineering professionals cause a constant scarcity of experienced people in developing countries. Despite several limitations, the software industries of some developing countries are continuing to emerge by catering to the majority of the local market as well as providing off-shore development centers for global companies.

A study presents the software development ecosystems in Malaysia (Baharom et al., 2005) which can be considered an emerging country in this context. However, the study did not consider the diverse development practices (e.g., security and scalability in development practices). There is also no study on a systematic comparison of the software engineering practices between developing and developed countries. In this paper, we focus on understanding the software development practices in an emerging software industry of another developing country, Bangladesh. Like Malaysia, Bangladesh is also a rapidly growing economy with 160 million population. IT sector is considered a priority sector in Bangladesh over the last decade. The software development industry dominates this sector. According to the Bangladesh Association of Software and Information Services (BASIS), 1100+ software companies operate in Bangladesh, where around 40% have a global business. The foreign revenue earned by the industry is over 800 Million USD (Basis, 2018).

Our study has four steps. First, we conduct a series of semi-structured interviews of eight software engineering professionals from four leading software companies in Bangladesh. We corroborate the interview findings with findings from similar studies of other countries like Canada, Turkey, Netherlands, New Zealand, etc. (Garousi and Zhi, 2013; Garousi et al., 2015; Vonken et al., 2012; Wang and Galster, 2018). The purpose is to gain an overview of the overall development practices and challenges. Second, we designed a survey based on the insights gained from the interviews. A total of 137 software practitioners from diverse software companies in Bangladesh responded to the survey. Third, we analysed the survey responses to understand the software development practices and challenges in Bangladesh. Fourth, we compare our findings against findings from other countries. We answer two research questions:

RQ1 What are the software development practices in an emerging country like Bangladesh?

RQ2 How do the development practices in Bangladesh differ from other countries?

We report the practices (RQ1) and the comparisons (RQ2) along four dimensions (D) as follows. The four dimensions were previously used in similar country-based studies, which thus helped us to compare our findings against their findings.

D1 *Software development methodologies used:* we investigate the development approaches, methodologies, and requirements analysis processes. We find that the software companies in Bangladesh mostly follow the agile methodology. In comparison with other countries, Bangladeshi software companies generally spend more time on the implementation stage of development. Technologically advanced countries spend more on system design (Cusumano et al., 2003; Groves et al., 2000).

D2 *Software tools and techniques used:* we determine the trending technologies like technology platforms, programming languages, frameworks, etc. We find that web-based software services are prevalent in the Bangladeshi software market and JavaScript is the most used language for web development. The degree of technologies and tools usage varies from the developed countries due to the availability of experienced developers, budget, etc.

D3 *Software testing and DevOps practices used:* we explore the present situation of testing and deployment practices adopted by the software firms in Bangladesh by asking questions about testing and deployment tools, test automation level, version control system, etc. We find that the usage of test automation and deployment tools is not widespread in the Bangladeshi industry. We also see that compared to developed countries, the test automation tool adoption is inadequate.

D4 *Performance and security measures used:* we analyse how software companies secure and maintain their code and what practices are followed to ensure performance and scalability. The responses show that standards are followed for security assurance, tools are used for performance testing, and scalability is mostly ensured by using cloud services. However, the companies in Bangladesh lag behind in the area of automated performance testing and automated deployment or continuous integration tools that might ensure resource-optimised scalability.

The findings can help software practitioners improve software engineering (SE) practices in Bangladesh or other developing countries with similar characteristics. The comparison with the developed countries shows the particular avenues for improvement in the emerging countries. For example, the limitation in using automated security and performance testing tools indicates the need for affordable security and performance profiling services SE industry in emerging countries.

1.1 Paper organisation

Section 2 presents the related work to our study. Section 3 describes the background of our study and the data collection procedure. Section 4 answers the two research questions. Section 5 discusses the implications of our findings. Section 6 discusses the threats to validity. Section 7 concludes the paper.

2 Related works

This section presents the related works that focus on software development practices and processes globally and in specific countries and regions.

2.1 Studies of development practices and processes

Cusumano et al. (2003) have conducted a global survey to identify software engineering practices. They have found that detailed architectural design and documentation is a common practice worldwide except in the USA. In the USA, only 32% of projects used detailed design specifications. One of the interesting findings of their study is the completeness of design before coding negatively correlates with the number of defects.

AlSubaihin et al. (2019) have identified the influence of the app store on software engineering practices. They have found that the perception of quality is slightly different among app store developers. App Store developers gave more priority to user rating than the traditional metrics like code quality and documentation when measuring software quality.

To identify the state of the practices in start-up companies, Klotins et al. (2018) have conducted a study on start-up companies. They found that start-ups apply market-driven requirements engineering instead of the standard software engineering requirement engineering. However, the applied requirements of engineering practices are often rudimentary and lack alignment with other knowledge areas (such as design).

2.2 Related region-specific studies

In 2012, Vonken et al. surveyed Dutch software-producing organisations to determine whether there is a gap between the current state of the practice and the state of the art in software engineering. From 99 respondents, they extracted 22 interesting observations. These observations mark insights into the development process that they found unusual or surprising, at least from an academic perspective. This unusualness could either stem from certain principles being applied less or more frequently than expected or from unexpected correlations observed between factors.

The survey conducted by Garousi et al. (2015) studies Turkey's software practices to characterise and understand the state of its SE practices. The military and defense software sectors are quite prominent in Turkey, especially in the capital Ankara region, and many SE practitioners work for those companies. 54% of the participants reported not using any software size measurement methods, while 33% mentioned that they had measured lines of code (LOC). In terms of effort, after the development phase (on average, 31% of overall project effort), software testing, requirements, design, and maintenance phases come next and have similar average values (14%, 12%,12%, and 11% respectively). Respondents experience the most challenges in the requirements phase. As a rather old but still widely used life-cycle model, the waterfall is the model that more than half of the respondents (53%) use. The next most preferred life-cycle models are incremental and agile development models with 38% and 34% usage rates, respectively. The waterfall and agile methodologies have slight negative correlations, denoting that if one is used in a company, the other will be less likely to be used. A recent survey conducted by Wang and Galster (2018) in 2018 shows that New Zealand

professionals use similar methodologies as professionals in other countries. Key findings of their study are:

1 popular programming language in the New Zealand software industry does not match with the worldwide ranking of popular languages

2 most of the time in SDLC is spent on implementation-related activities rather than analysis and design.

In another study, Groves et al. (2000) reported that the New Zealand software industry pays particular attention to requirements gathering. They surveyed a selection of software companies with a general questionnaire and then conducted in-depth interviews with four companies. They found a clear difference in the testing phase between large and small software companies. Their finding is larger companies pay more attention to testing than smaller companies.

The study conducted by Sison et al. (2006) presents an exploratory survey and case study results on software practices of some software firms in five ASEAN countries (Malaysia, Philippines, Singapore, Thailand, and Vietnam). They found that most of the firms in that region do not follow the standard procedure for SQA. In a study focusing on the test practices in Canadian firms, Garousi and Zhi (2013) found that the number of passing user acceptance tests and the number of defects found per day are considered the most important quality assurance metrics in Canadian firms. They compared their result to a previous study and showed that Canadian firms are giving more importance to testing-related training than in the past.

Baharom et al. (2005) conducted a study on Malaysian software firms to find the effectiveness of standard practices. They found that alpha and beta testing was hardly implemented in software firms. Another interesting finding of their study is that most Malaysian firms emphasise implementation; only a negligible number of companies spend more than 20% of the effort in planning and design. Zafar et al. (2018) have surveyed why Pakistani software firms do not follow the standard requirement engineering process. They have found multiple factors contributing to the issue, such as lack of budget, lack of time, lack of dedicated team, etc. However, the most prevalent issue is lack of budget; more than 60% of their respondents have responded that the standard requirement engineering process was not followed due to scarcity of budget.

To identify the software engineering practices in Bangladesh, Rahim et al. (2017) have surveyed 41 practitioners in the Bangladesh software industry. One of their interesting findings is the waterfall model is still popular among them. They found that 40% of respondents indicated that requirement analysis and prioritisation are the most challenging software development processes. In another survey focusing on testing practices in the Bangladesh software industry, Bhuiyan et al. (2018) found that most companies do not follow any standard SQA techniques for their projects. The interesting fact is such malpractice does not hinder their progress; they reported that these firms have been in the industry for 6.5 years on average. However, in another survey, Begum et al. (2009) found that 47.5% of respondents follow standard SQA techniques in their projects.

Based on a 200 participants survey, Hussain et al. (2020) concluded that the computer science undergraduate education system in Bangladesh leaves most of its graduates unprepared for the software industry. They suggested that updating the

syllabus as part of the curriculum and including internships could help make graduates fit for the industry.

3 Study setup

Our study has four steps as follows:

Step 1	*Interview (Subsection 3.1):* we conduct a series of semi-structured interviews to understand the current software engineering practices in Bangladesh and explore how such practices could differ from other countries.
Step 2	*Survey (Subsection 3.2):* we design survey questions to get a deep understanding of the insights collected from the interviews.
Steps 3, 4	*Data analysis to answer RQ1 and RQ2 (Subsection 3.3):* we analyse the survey responses to answer RQ1 (understand development practices in Bangladesh) and RQ2 (compare the practices against other countries).

3.1 Interview

The goal of the interview session was to prepare the survey questions. Eight individual participants from four leading software companies were interviewed. First, we designed an initial list of survey questions in Google form by consulting previous studies in other countries like Canada, Turkey, Netherlands, New Zealand, etc. (Garousi and Zhi, 2013; Garousi et al., 2015; Vonken et al., 2012; Wang and Galster, 2018). Each participant was asked to identify ambiguities in the question. From the feedback of the interview session, we revised the questions. Each interview session lasted about half an hour. Interviewees were first asked to complete the survey. After completing the survey, we asked what he/she understood from the questions and what he/she meant by the answers. Throughout the interviews, we identified discrepancies between the understanding of the interviewee and the goals of the survey questions by comparing the interview findings against findings from related work. The primary purpose of the interview was to validate the questionnaire with experts, and then we revised our survey questions accordingly before distributing it for the survey.

3.2 Survey participants

The survey questions are shown in Table 1. There are 17 questions, 14 closed and three open-ended. We targeted developers who are currently working in the software industry of Bangladesh. We applied purposive sampling (Vogt and Johnson, 2005) to include respondents in a software development-related role. Purposive sampling is basically based on the assumption of the population. It is possible that some elements will not have a chance of selection in this method. Moreover, the probability of selection cannot be accurately determined in this process. We shared the survey link through the authors' personal connection and in the local developers' groups on social media to achieve our sampling goal. We also implemented the chain referral strategy (Creswell,

2013) and asked others to pass on the survey invite. Due to such a snowball approach to recruiting survey participants, it is not possible to calculate the response rate of our survey. We have conducted the survey through Google Forms. The survey link was opened before the invitations were sent. The survey link was open for feedback for about two months and was closed when two consecutive weeks were found without any response. In total, we received 137 responses from the survey. Each participant was first asked a series of demographic questions (e.g., roles, experience, gender) and then presented the survey questions related to the development practices. The respondents were able to check more than one option in all closed-ended questions. Table 2 shows the distribution of the survey participants by their roles. We noticed that a significant number (69%) of our respondents are developers. Since software developer/developer is a generic role, this can be noticed in other surveys on the SE industry, like the Stack Overflow survey (Stack Overflow, 2020, 2019). Previously conducted studies on the Canadian (Garousi and Zhi, 2013) and Turkish (Garousi et al., 2015) SE industry found that more than 80% of respondents were developers. Other roles for respondents to our survey are managers (16.9%) and other kinds of software engineers (8%) (e.g., data engineer, R&D engineer). Although the studies in Canadian (Garousi and Zhi, 2013) and Turkish (Garousi et al., 2015) SE industry did not report the presence of DevOps developers, we assume DevOps and testing professions are bundled under the software quality assurance (SQA) engineering profession in the two studies.

Table 1 Survey questions (without demographic) along the four dimensions (D)

D1	*Software development methodologies used*
$6_{127,C}$	Which of the following software development methodologies do you follow?
$7_{120,C}$	Which of the following do you use for requirements gathering?
$8_{126,C}$	On which software development activities do you spend most of the time?
D2	*Software tools and techniques used*
$9_{128,C}$	Which of the following technologies do you have experience working in?
$10_{127,C}$	What is the primary operating system you are developing on?
$11_{128,C}$	Which programming languages are you using?
$12_{109,C}$	Which frameworks are you using?
$13_{125,C}$	Which IDE are you using?
D3	*Software testing and DevOps practices used*
$14_{117,C}$	What types of software testing practices do you use?
$15_{118,C}$	What is the level of automated testing in your projects?
$16_{83,C}$	Which tools do you use for testing and quality assurance?
$17_{60,C}$	Which tools do you use for continuous deployment?
$18_{121,C}$	Which version control tool do you use?
D4	*Security and performance measures used*
$21_{74,O}$	How do you ensure scalability of your products?
$22_{73,O}$	How do you maintain performance of your products?
$23_{74,O}$	How do you ensure security of your products?

Note: Subscripts with a question number show # of responses and question type (O = open, C = closed).

Table 2 Role-wise distribution of participants

Role of the participants	*Percentage*
Developer	69.72%
Manager	16.9%
SQA engineer	7.04%
Business analyst	1.4%
R&D engineer	1.4%
Data engineer	0.7%
Software architect	0.7%
Team lead	0.7%
Trainer	0.7%
UX designer	0.7%

Table 3 Experience-wise distribution of participants

Experience of the participants	*Percentage*
Less than 2 years	33.58%
2 to 5 years	24.82%
5 to 10 years	18.98%
More than 10 years	17.52%
Experience not disclosed	5.11%

Table 3 shows the distribution of the survey participants by their experience. More than 61% of the participants worked in the industry for at least two years. In terms of work experience, the demographics of our survey are similar to previous surveys. 77% of our respondents have less than ten years of experience. In the Canadian and Turkish SE industry surveys, the percentage of respondents with less than ten years of experience is 67.9% and 79%, respectively.

About 59% of the respondents disclosed companies they work for. We have received responses from 38 different companies that cover a large space of software developers. About half of the companies (51.9%) are involved in web development, application development, and ERP software development. However, the companies offer a variety of products such as IoT-based health monitoring, cloud services, telecom, ride-sharing platform, biometrics-based personal identity management, and security solutions. According to the Bangladesh Association of Software and Information Services (BASIS), about 1,100 software farms are employing 300,000 IT professionals. Around 40% of our respondents are from the top 10 software firms in terms of revenue tax listed by BASIS. The remaining 60% of the respondents are from the 23 other software companies focusing on all the major sources of software products developed in Bangladesh (e.g., mobile, web, B2C, B2B, etc.).

3.3 Data analysis to answer RQ1 and RQ2

The survey consisted of both closed and open-ended questions. We analysed the closed questions using standard descriptive and statistical techniques. We analysed the closed questions following the principles of open coding. Open coding includes labeling of

concepts/ categories in textual content based on the properties and dimensions of the development entities (e.g., tools, processes, etc.) about which the contents are provided. A systematic qualitative data analysis process was followed to analyse the open-ended questions. First, the two authors independently coded the initial 30% responses to each question to extract potential categories. Second, the authors conducted discussion sessions to develop a unified common coding scheme for each question using these categories. Third, the rest of the responses were coded using this coding scheme using the coding analysis toolkit (CAT) (Lu and Shulman, 2008) software. To measure the level of agreement between two coders, we used the online tool Recal2 (Freelon, 2020) and CAT (Lu and Shulman, 2008). The Recal2 calculator reports the agreement using four measures:

1 percent agreement

2 Cohen κ (Cohen, 1960)

3 Scott's π (Scott, 1955)

4 Krippendorff's α (Krippendorff, 2004).

It is believed that Krippendorff's α is more sensitive to bias introduced by a coder and is recommended (Joyce, 2013) over Cohen's κ (Cohen, 1960).

The level of agreement between the first two coders is presented in Table 4. The average κ value was 0.71 and Krippen α value if 0.73. It is a common practice that a κ value between 0.61 and 0.80 (Landis and Koch, 1977) is considered a 'substantial agreement.' In the coding process, a large number of codes were generated from each of the open-ended questions. To help with our analysis, we conducted discussion sessions to identify the codes that express similar themes.

After reaching a consensus, we grouped those codes into a smaller number of high-level categories. We have used the statsmodels (Seabold and Perktold, 2010) and scipy (Virtanen et al., 2020) modules in Python for statistical analysis like hypothesis testing to check differences between two categorical distributions. In particular, we have used the following two standard statistical measures for hypothesis testing during our data analysis:

- *Chi-squared test of independence* is used to identify the difference between observed and expected frequencies. For example, we have observed that mobile apps and web platforms mostly use Java and JavaScript as programming languages. To check our hypothesis of whether the development platform and choice of developing language are related, we conducted the chi-square test of independence.
- *Mann-Whitney U test* is used to compare between two groups. For example, in this study, we have used Mann-Whitney U test to check whether there is a significant difference in Linux use among experienced developers and junior developers groups.

Both chi-squared and MW tests are non-parametric, which are suitable for comparing distributions in survey data that are not normally distributed. Depending on the type of distribution, we may either use one of the tests at a time. For example, if we simply have the total count of a category (e.g., types of developers using Java vs. Python),

we can use the chi-squared test of independence to determine whether the usage of the two languages between two developer types (e.g., novice vs. expert) is statistically significant. However, if have counts of usage of a language (e.g., Java) over time for the two developer types, we can use Mann-Whitney U test to check if the distribution is statistically significant or not.

Table 4 The agreement level in the open coding (#QT denotes # quotes used)

	Q21 (#QT = 74)	*Q22 (#QT = 73)*	*Q23 (#QT = 74)*
Percent	76.6	72.3	73.7
Cohen κ	0.731	0.688	0.71
Scott π	0.731	0.688	0.71
Krippen α	0.732	0.689	0.711

4 Study results

We answer two research questions by analysing our survey responses and by comparing the results against other countries:

1 What are the software development practices in an emerging country like Bangladesh? (Subsection 4.1)

2 How do the development practices in Bangladesh differ from other developed countries? (Subsection 4.2)

The first research question focuses on the dimensions of software development practices and processes in Bangladesh. The second research question compares the practices prevailing in Bangladesh with those of other regions to reveal the similarities and dissimilarities in software development practices and processes.

4.1 Development practices and processes in Bangladesh (RQ1)

Following previous studies in region-based software development process analysis (Garousi and Zhi, 2013; Garousi et al., 2015; Vonken et al., 2012; Wang and Galster, 2018), we analyse the development practices and processes used in Bangladesh along four dimensions (D):

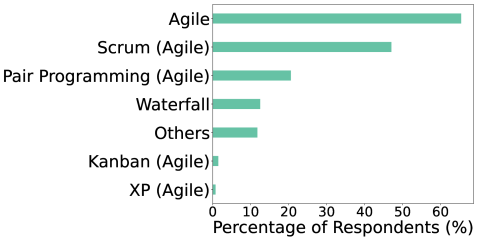
1 software development methodologies (Subsection 4.1.1): we investigate development processes, methods, and requirements analysis processes

2 software tools and techniques (Subsection 4.1.2): we determine trending technologies like platforms, programming languages, and frameworks

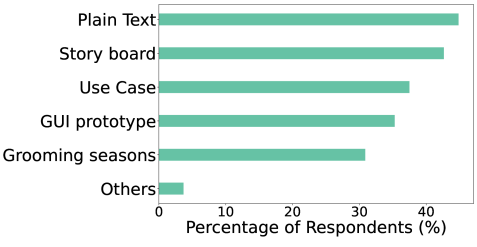
3 software testing and DevOps practices (Subsection 4.1.3): we explore testing, deployment, and continuous integration practices

4 performance and security measures (Subsection 4.1.4): we analyse how software companies secure and maintain their code and ensure overall system performance and scalability.

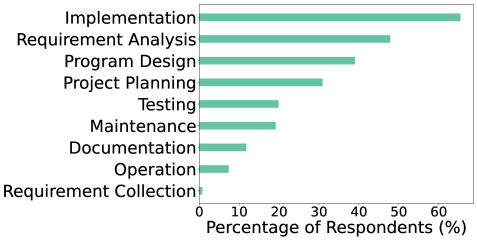
Figure 1 Software development methodologies used by the respondents, (a) software development methodologies (Q6) (b) requirements gathering (Q7) (c) timeline of development activities (Q8) (see online version for colours)



(a)



(b)



(c)

4.1.1 *Software development methodologies used (D1)*

Software development methodology implies the process used for developing a particular software in a structured and methodical way. We asked three questions relevant to this investigation as follows: software development methodologies (Q6), requirements gathering process (Q7) and most time-consuming software development activities (Q8).

- *Software development methodologies (Q6):* when the respondents were asked to provide the methodologies they use in development, the replies included the pure agile approach along with the different variations of the agile methods (e.g. scrum, kanban, XP, etc.). Agile is the most popular (64%) development methodology in Bangladesh, followed by its variation scrum (46%). These outcomes match the 2018 Stack Overflow survey (Stack Overflow, 2018), which also reports that agile and scrum are the most popular methodologies worldwide. Among the other methodologies, pair programming (20%) and waterfall (12%) are also popular in Bangladesh software companies.

- *requirements gathering process (Q7):* one of the most critical activities of the software development life cycle is collecting and analysing the requirements of a system. Usually, the outcome of the analysis is presented before the client and modified according to their feedback. The clearer and more detailed requirements are, the higher the possibility of building software that conducts the client's anticipation. Corresponding to Figure 1(b), using plain text (44%) and storyboard (41%) are the most widely used requirement-gathering techniques among our survey participants. The other relevant techniques include use case (36%), GUI prototype (35%), grooming session (30%), etc.
- *Timeline of development activities (Q8):* according to 65% of our respondents, most of the time is spent in the implementation stage, whereas the requirement analysis stage requires the second most according to 45% response. The other usages are program design (37%), project planning (30%), testing (19%), maintenance (17%), etc. We have conducted a two-way chi-squared test of independence to check whether the software development methodologies and the most time-consuming activities are associated or not. We have received $p = 0.60$, which means there is not enough evidence in the data to claim that software development methodologies and the most time-consuming activities are related.

RQ1-D1. Software development methodologies used: To provide software services, companies prefer the agile approach followed by scrum and also collect requirements via plain text in a high percentage. Besides, developers of the Bangladeshi SE industry generally spend more time on implementation-related activities than planning and testing.

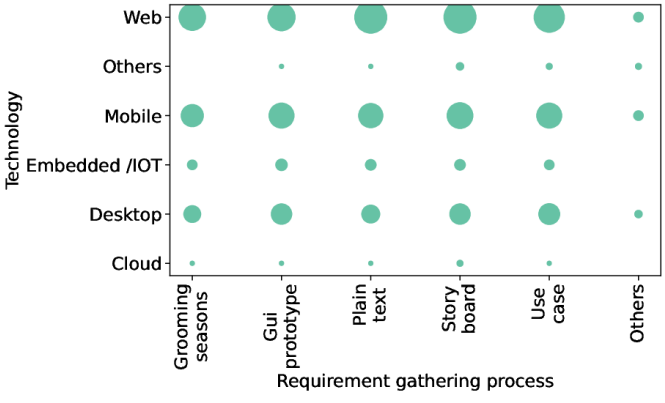
4.1.2 Software development tools and techniques used (D2)

We ask five questions related to the adoption of technologies and tools by the participants: technology platform (Q9), operating system (Q10), programming language (Q11), framework (Q12) and IDE (Q13).

- *Technology platforms (Q9):* 80% of survey respondents work for the web platform [Figure 3(a)]. The rest are engaged in mobile (45%), desktop (30%), and embedded/IoT (8%) development. This distribution is similar to the 2020 worldwide survey of JetBrains (JetBrains, 2020), which finds that websites are the most common types of application that the developers work on, and the web platform is the most preferable and popular to develop, followed by desktop and mobile. We have conducted a cross-aspect analysis to identify any relationship between the technology platform and the requirement-gathering process. The bubble charts in Figure 2 visualise the cross-aspect analysis. It is clear from the figure that the requirement-gathering process is mostly practiced in GUI-based development (e.g., web, desktop, mobile).
- *Operating systems (Q10):* most of our respondents preferred Linux-based operating system (56%) for their development [Figure 3(b)]. The second most frequently used operating system is Windows (45%) followed by MacOS (28%).

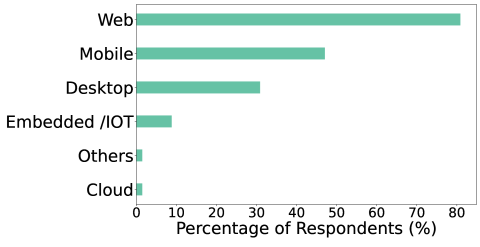
We observed similar scenarios in the 2018 and 2019 StackOverflow (SO) surveys (Stack Overflow, 2018, 2019). However, Windows was ranked first in the 2020 survey of both SO and JetBrains (Stack Overflow, 2020; JetBrains, 2020). The recent higher preference towards Windows could be due to the newly included WSL (Windows subsystem for Linux) that allows users to perform almost any Linux-specific task on Windows. We anticipated that the use of OS might be related to professional experience. Among the participants, senior/expert developers (those with at least 5 years of experience) have significantly higher rates of Linux usage ($p = 0.024$ based on the Mann-Whitney U test).

Figure 2 Cross aspect analysis of requirement gathering and technology platform (see online version for colours)

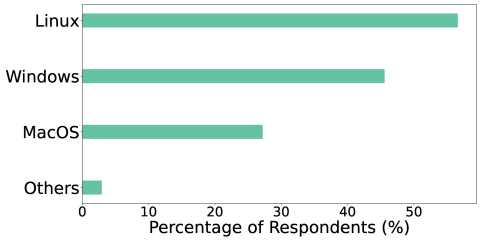


- *Programming languages (Q11):* around 65% and 60% of our respondents use JavaScript and Java respectively, which are the two most used languages in Bangladesh [Figure 3(c)]. Both JavaScript and Java are popular for web and mobile platforms. A great percentage of our survey participants develop for both web and mobile platforms. Other languages like PHP (25%), Python (25%), and C# (18%) are also used, which indicates that the software engineers are not biased towards a single specific language. Our survey result matches with the last two years Stack Overflow survey and the GitHub stat. In all of the cases, JavaScript is the most used language, followed by Java and Python (Stack Overflow, 2020, 2019; Github, 2020). We observed that users using mobile and web platforms mostly use Java and JavaScript as their programming language. However, the observation is not statistically significant ($p = 0.1$). From the chi-square test of independence, we have found that the development platform and choice of programming languages are not associated. Though the use of the operating system can be influenced by programming language (e.g., Swift and macOS), we do not find any significant correlation between the two choices.

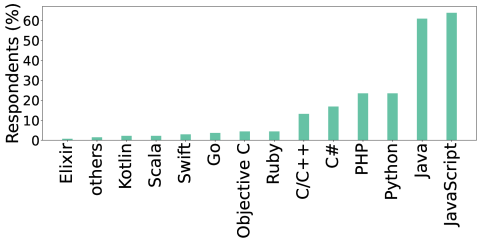
Figure 3 Software development tools and techniques used by the respondents, (a) technology platforms (Q9) (b) operating systems (Q10) (c) languages used in software development (Q11) (d) frameworks used in development (Q12) (e) IDEs used by the respondents (Q13) (see online version for colours)



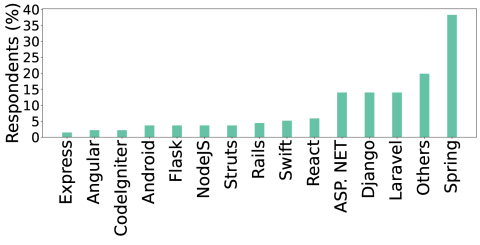
(a)



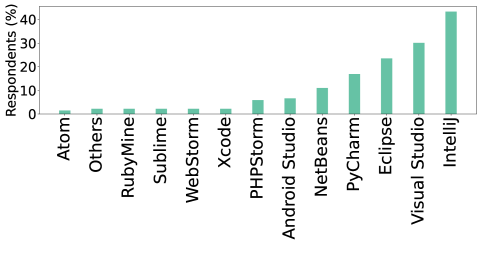
(b)



(c)



(d)



(e)

- *Frameworks used in development (Q12):* as shown in Figure 3(d), Spring Boot (37%) is the most used framework in the Bangladesh software industry. This observation is aligned with the result of Java's usage rate corresponding to Figure 3(c). Since JavaScript is the most used language of our respondents, they use various JavaScript frameworks such as React, Node.js, Angular, Express, etc. ASP.NET, Django, and Laravel are all used by around 15% of our respondents. React, Swift, Ruby on Rails, Node.js, etc., are comparatively less used. Other than these, lots of frameworks such as Cocoa, Meteor, TestNG, Relay, Appium, CakePHP, etc., are also used [presented as 'Others' in Figure 3(d)]. For web development, Django and Spring frameworks are mostly used in Bangladesh. We have compared our results with the Stack Overflow 2016 to 2020 survey (Stack Overflow, 2017, 2018, 2019, 2020). The only common framework in the top five list in both surveys is ASP.NET. In the Stack Overflow survey, we noticed that JavaScript-based frameworks (e.g., Jacqueline, Angular, React, Node.js) occupy the top positions (top five), which is not the case for our survey.
- *IDEs used by the respondents (Q13):* as shown in Figure 3(e), IntelliJ is used by the highest number of respondents (43%). IntelliJ is an integrated development tool for developing software using Java for enterprise, mobile, and web applications. The other IDEs used in SE industries are Visual Studio (30%), Eclipse (24%), PyCharm (17%), NetBeans (11%), and Android Studio (7%).

RQ1-D2. Software development tools and techniques used: Web-based software services top the list of development technologies. The requirement-gathering process is mostly practiced in GUI-based development. Practitioners prefer Linux-based operating systems (OS) mostly, though other OSs (e.g., Windows, macOS) also have an appreciable usage rate. JavaScript and Java are the two most popular languages. Hence, Java integrated tool, IntelliJ tops the list of IDEs followed by Visual Studio and Eclipse.

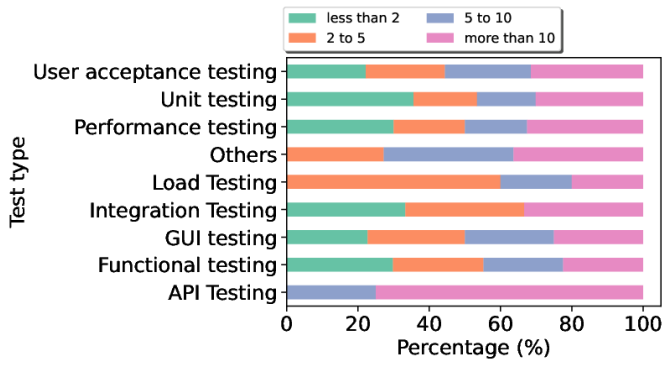
4.1.3 Software testing and DevOps practices used (D3)

We asked five questions to determine the adoption of testing and DevOps techniques in Bangladesh software companies: software testing practices (Q14), level of automated testing (Q15), tools used in testing and QA (Q16), continuous deployment tools (Q17) and version control (Q18).

- *Software testing practices (Q14):* according to Figure 6(a), several testing practices are used during software development. The results show that most of the organisations have carried out unit testing (53%), functional testing (49%), user acceptance testing (39%), GUI testing (31%), etc. Unit testing also ranked first in the 2019 survey of JetBrains, where it was voted by 71% participants across the globe (JetBrains, 2019). We observed in our survey that in some cases managers have reported performing GUI testing and performance testing, which is unlikely in their role/designation. It may be deduced that in the absence of enough specialised resources, managers have to take additional responsibility. To identify the relation between testing practices and experience, we plotted them together in Figure 4. We have observed that junior developers (less than 5 years of experience) mostly perform unit, integration, and functional testing, whereas

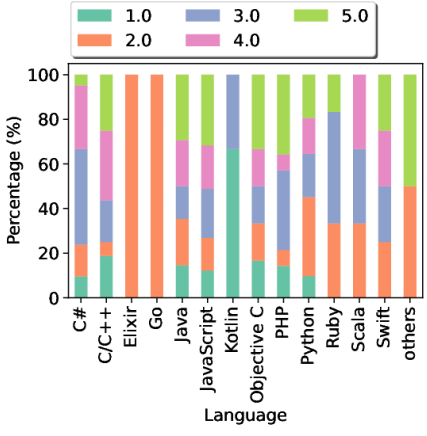
senior developers mostly perform API testing. We conducted the Mann-Whitney U test to assess the conjecture, and it was found statistically significant ($p < 0.01$).

Figure 4 Testing practices and professional experience (see online version for colours)

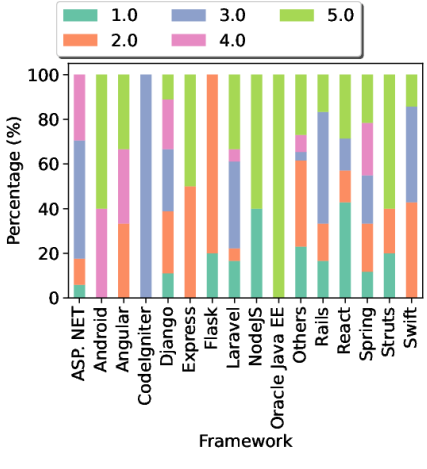


- *Level of automated testing (Q15):* we have asked the survey participants about the level of automated testing performed in their respective companies. The responses were gathered using the Likert scale. It was found that different respondents have very different experiences in this context, i.e., some companies heavily practice automated testing, while others favor manual testing. Results are shown in Figure 6(b), which indicates that about 70% of our respondents (others than those who voted for level 5) do not use automated testing regularly. The level of automated testing might be related to the programming language/framework. The testing suite provided by the framework/language might encourage developers to implement automated testing. The level of automated testing vs. language and framework is plotted in Figures 5(a) and 5(b), respectively. It seems from Figure 5(a) that the highest level of automated testing is mostly practiced in Java, JavaScript, Objective-C, and PHP language. We conducted the Mann-Whitney U test to assess our conjecture, and it was found statistically significant ($p = 0.01$). From Figure 5(b), we found that the highest level of automated testing is mostly performed in Android, Express, Node.js, Struts, and Java EE framework, and the observation is statistically significant ($p = 0.006$ based on chi-square test of independence). Also, the highest level of automated testing is mainly used by developers (unit testing). Managers practice the lowest level of automated testing. We observed that managers are mainly engaged in assessing the acceptability of the product from the end-user's point of view. We also observed that junior developers tend to use more automated testing than senior developers. However, the observation is not statistically significant; the experience and automated testing level are not associated ($p = 0.08$ based on the chi-square test of independence). One reason behind this observation may be that the senior developers perform GUI testing more, which is hard to automate.

Figure 5 Analysis of automated testing level, (a) programming language and automated testing level (b) framework and automated testing level (see online version for colours)



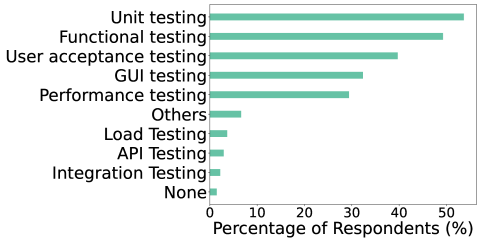
(a)



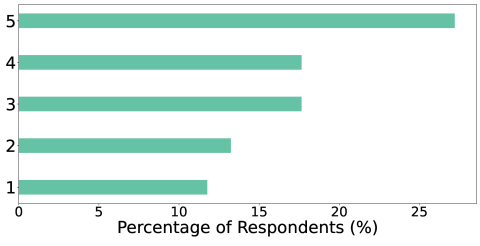
(b)

- *Tools used in testing and QA (Q16):* most of the respondents have used XUnit (e.g., JUnit, NUnit) (30%), Selenium (27%), Jenkins (20%), others (9%) [see Figure 6(c)]. Around 38% of our respondents were not interested in replying to this question. This is not surprising because the majority of the respondents (93% approx.) were working on roles other than SQA engineer as per Table 2, and they are not supposed to be involved in any testing other than unit testing themselves.

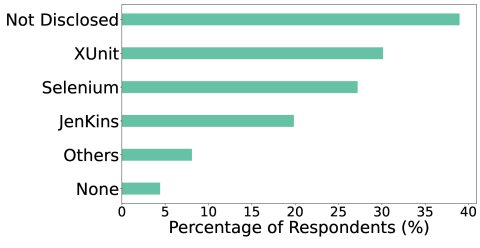
Figure 6 Software testing and DevOps practices followed by the respondents, (a) software testing practices (Q14) (b) level of automated testing (Q15) (c) tools used in testing and QA (Q16) (d) continuous deployment tools (Q17) (e) version control (Q18) (see online version for colours)



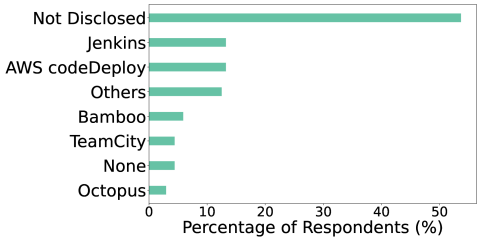
(a)



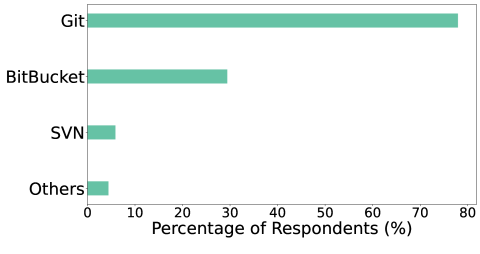
(b)



(c)



(d)



(e)

- *Continuous deployment tools (Q17):* majority of the respondents deploy their implemented codes using AWS code-deploy (12%) and Jenkins (12%) (Figure 6(d)). The other deployment tools are Bamboo (5%), TeamCity (4%), Octopus (2%), etc. 4% respondents confirmed that they did not use any deployment tools and 53% of the respondents were not interested in this topic. However, the percentage of uninterested respondents does not seem unexpected. From Tables 2 and 3, we can observe that a significant portion of our respondents are developers, and more than half of our respondents are experienced for less than five years. As deployment is related to DevOps, it is quite likely that developers, especially junior ones, do not have exposure to the DevOps process or have less interest in deployment. The outcome indicates that the usage rate of deployment tools in Bangladesh for continuous integration and continuous delivery is not widespread yet.
- *Version control (Q18):* Git (78%) and Bitbucket (29%) are mostly used version control systems in the software industry.

  Besides these, Subversion (SVN) (5%) and others (4%) are used. Respondents were allowed to select more than one option. The 2018 Stack Overflow survey (Stack Overflow, 2018) reports that the most popular version control system is Git (87.2% developer uses Git) and the second most popular is SVN (16.1% developer uses SVN).
  item [] However, in our survey, we found a slightly different result, the most popular version control system is Git and the second most popular is Bitbucket. This might be related to the declining popularity of SVN over the years. From the Stack Overflow survey over the range 2017–2018, it is clear that SVN is losing popularity to Git.

RQ1-D3. Software testing and DevOps practices used: Unit testing is heavily carried out by developers in the Bangladesh SE industry likewise across the globe. In addition, various software testing tools (e.g., XUnit, Selenium, etc.) and deployment tools (e.g., AWS code-deploy, Jenkins, etc.) are used. However, there is a tendency among most Bangladeshi companies of not use these automated tools regularly. Globally popular container-based technologies like Docker and Kubernetes are seldom used in the Bangladesh SE industry during the timeline of this survey, but few companies have started using them recently. To take advantage of the ubiquitous cloud-based deployment, the Bangladesh SE industry should focus more on adapting these technologies.

4.1.4 Performance and security measures used (D4)

Security and performance are two of the most important non-functional requirements for any software product. We asked three open-ended questions with regard to the enforcement of security and performance-related standards in the software products in Bangladesh: how do you ensure performance, scalability (Q21, Q22), and security (Q23) in your software products?

- *Performance (Q21, Q22):* software performance indicates how efficient the software is in terms of response time and resource consumption. We find 12 types of performance measures that are practiced in software products developed in

Bangladesh (see Figure 7). The 12 types are divided into four categories: use of tools and frameworks (46.72%), design principles/best practices (26.17%), testing (16.82%), review and feedback (9.35%) and database optimisation (3.74%).

1 *Use of tools and frameworks:* six out of 12 types belong to this category.

   a *Tools:* around 18.69% respondents use tools and metrics to measure performance: "take help of different performance monitoring tools and dashboard, analysed data, measure time and memory efficiency of process" (S_{35}).

   b *Infrastructure:* to ensure performance, around 20.56% of respondents use upgraded infrastructure such as cloud hosting (e.g., Amazon AWS), high-end servers, and new technologies.

   c *Caching:* around 2.8% respondents implemented caching to maintain software performance.

   d *Container technology:* containers enable users to scale their system without any dependency on the underlying OS. About 1.87% of our respondents use container technologies to ensure their products' scalability and performance.

   e *Using SDK/framework:* about 0.93% of respondents depend on the framework to maintain software performance and scalability.

   f *Load balancing:* around 1.87% respondents use load balancing as a measure to maintain performance: "optimizing number of HTTP requests, asynchronous programming, caching, CDN, load balancing, Nginx, varnish, compression of data, continuous monitoring, load testing, stress testing" (S_{42})

2 *Use of design principles/best practices:* around 26.17% of respondents try to ensure software performance right from the design phase as follows:

   a *Using better codes/practices:* around 9.35% of respondents ensure performance by implementing industry-standard best practices like compression technology, enforcing design patterns, and refactoring.

   b *Efficient designing:* around 16.82% of respondents emphasise on performance-aware architecture design.

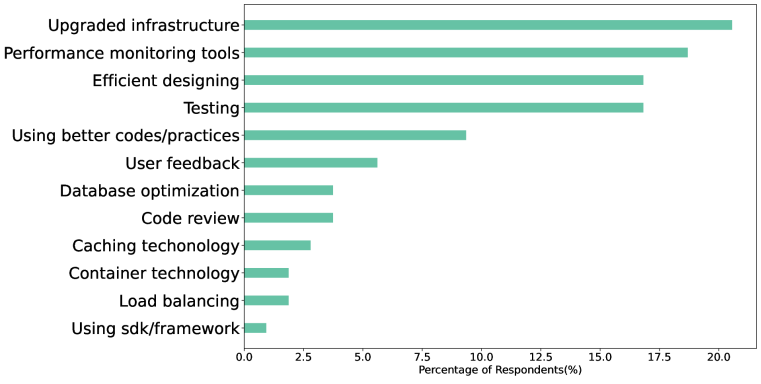
3 *Use of testing:* around 16.82% respondents rely on the software testing strategy to ensure performance like load testing and stress testing.

4 *Use of review and feedback:* around 9.35% of our respondents use user feedback (e.g., continuous feedback from QA team S_{65}) and code review to improve product performance. According to S_{15}: "the code quality is assessed by the different team members during code review, followed by designing new ways to solve issues in the product that are time-intensive".

5 *Database optimisation:* around 3.74% of our respondents use database optimisation to ensure performance and scalability. Database optimisation

includes sharding, clustering, indexing, and scaling. According to S_{85}: "besides scaling horizontally, database scaling is performed by partitioning tables, along with multi-threaded implementations".

Figure 7 Measures to ensure performance and scalability of products (see online version for colours)



- *Security (Q23):* our open coding of the survey responses reveals 11 labels (see Figure 8). The 11 labels are divided into three main categories of security-related development practices: measures related to authentication and authorisation (64.82%), exploitation of tools and techniques to ensure security in products (53.71%) and use of encryption technologies for data (7.41%). We discuss the categories below.

    1 *Measures related to authentication and authorisation:* six out of the 11 labels belong to this category.

        a *Multi-prong standard process:* about 29.6% of the respondents reported that they practice various security standards and protocols to ensure security (e.g., ISO/IEC 27001, PA DSS).

        b *Token-based authentication:* about 13% respondents reported to have implemented a token-based authentication system, which allows users to enter their username and password to obtain a token for authentication and authorisation.

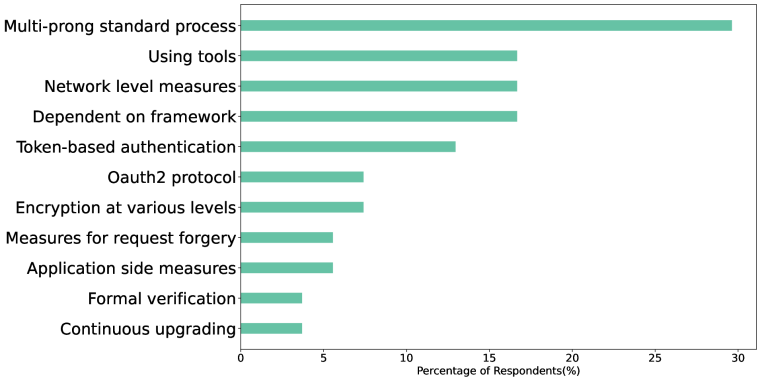
        c *OAuth 2.0:* around 7.4%) respondents use the OAuth 2.0 protocol as the primary way of maintaining security. OAuth 2.0 is the industry-standard protocol for authorisation. OAuth 2.0 focuses on client developer simplicity while providing specific authorisation flows for web applications, desktop applications, mobile phones, and living room devices.

        d *Application-side measures:* around 5.6% of respondents implement security measures at the application level like encryption of application data at the client side, use of HTTPS while pulling data from a server, secured architecture, etc.

e *Measures for request forgery:* around 5.6% respondents implemented security measures against cross-site request forgery (e.g., attacks like cross-origin resource sharing (CORS), cross-site request forgery (CSRF) or one-click attack or XSRF). Security testing is paramount for this: "security testings like: SQL injection, cross-site scripting, CSRF, API security, use of HTTPS, detecting malicious/suspicious HTTP requests and auto-blocking" (respondent ID S_{42}).

f *Formal verification:* around 3.7% respondents ensured the practice of formal code review to enforce security practices: "there are some basic guidelines that we must follow and while code review this needs to be an absolute part that needs to be checked before the code gets merged" (S_{112}).

Figure 8 Measures to ensure security of products (see online version for colours)



2 *Exploitation of tools and techniques to ensure security in products:* four out of the 11 labels belong to this category.

a *Dependent on framework:* around 16.7% respondents depend on the underlying framework for security like Spring, HDIV, and Laravel. "HTTPS, the popular framework which already prevents some types of attacks. rest of the things on case-by-case basis)" (S_{79}).

b *Use of tools:* respondents use various open-source/paid tools for scanning and testing like OWASP and penetration testing tools.

c *Network level measures:* Network-level measures include IP-white-listing, port-blocking, VPN, and the use of HTTPS in software. 16.7% of respondents use at least one of the mentioned strategies to ensure security.

d *Continuous upgrade:* around 1.5% respondents reported that they arrange frequent hackathons, workshops, and security audits: "we run security audits of our office environment. We also conduct security sessions per 6 months to introduce the latest trend in threats and what we can do to avoid them" (S_{57}).

3 *Use of encryption technologies for data:* around 7.4% respondents use encryption at the different levels of software architecture such as network, data, and transmission. "use encryption at different level of software (server, network, transmission layer, database and software layer)" (S_{35}).

RQ1-D4. Security and performance measures used: Bangladesh software industry uses various methods to ensure security, but the adoption of tools is not widespread. We have noticed that the Bangladesh software industry mostly uses performance monitoring tools and software testing to ensure product performance. Software scalability is generally considered at the design stage (e.g., efficient design) in the Bangladesh SE industry.

4.2 Comparison of development practices among countries (RQ2)

Given Bangladesh is a developing country with an emerging software development industry, our observations of Bangladesh SE industry can offer insights into how such an industry is operating in other similar countries and how such observations could differ from countries with matured SE industries. In this section, we offer a comparative assessment of our observations in the Bangladesh SE industry against observations reported previously in other countries like Canada, Turkey, Netherlands, New Zealand, etc. (Garousi and Zhi, 2013; Garousi et al., 2015; Vonken et al., 2012; Wang and Galster, 2018). Similar to RQ1, we present the comparisons along four dimensions:

D1 software development methodologies used (Subsection 4.2.1)

D2 software tools and techniques used (Subsection 4.2.2)

D3 software testing and DevOps practices used (Subsection 4.2.3)

D4 security and performance measures used (Subsection 4.2.4).

The summary of the comparison is presented in Table 5. Again, the summary of the comparison between developed and emerging countries is presented in Table 6.

4.2.1 Software development methodologies used (D1)

We compare the observations from three questions in our survey: software development methodologies (Q6), requirements gathering (Q7) and most time-consuming software development activities (Q8). The comparisons are discussed below:

- *Software development methodologies (Q6):* our study shows that the most acceptable method in Bangladesh is the agile model (64%) likewise across the globe (Stack Overflow, 2018). However, the usage of the scrum (44%) in New Zealand has better usage followed by agile (30%) (Wang and Galster, 2018), and in Turkey, the waterfall model is mostly used based on the survey of Garousi et al. (2015). Again, in both Bangladesh and New Zealand, extreme programming (XP) has a lower percentage of usage. Almomani et al. (2015) found that software developments in Malaysia are predominantly regulated through an ad hoc approach (53%) and the agile methodologies (46%) since usually software organisations are majorly concerned with short-term delivery of software products.

Table 5 Summary of the comparison of development practices among countries

Dimension	*Prior study*	*Comparison*
D1	Scrum and waterfall (Wang and Galster, 2018; Garousi et al., 2015) are the most used development methods. Though requirements are specified in text format in the Netherlands (Vonken et al., 2012), well-specified design documentations are practiced in Japan and Europe (Cusumano et al., 2003).	Like Malaysia an ad hoc-based software development practice exists in Bangladesh. Advanced industries give high importance to planning. However, in Bangladesh, the implementation phase gets the highest priority.
D2	Web (as tech platform) and Windows (as OS) are the dominating platforms in New Zealand (Wang and Galster, 2018). Although Java is a popular language in Turkey (Garousi et al., 2015), it is not that popular in New Zealand (Wang and Galster, 2018)	Like other countries web-based platforms have widespread demand in Bangladesh. Bangladesh SE industry has a similarity with Turkey in terms of the popularity of Java and Python.
D3	Unit testing is highly practiced in technologically advanced SE industries (Garousi and Zhi, 2013; Wang and Galster, 2018). However, the level of test automation is not satisfactory in the European SE industry (Dutta et al., 1999).	In Bangladesh, unit testing is less extensive than the advanced SE industry practice. However, it has similarities with the European SE industry in test automation.
D4	Though it is a common practice in advance industries security (Garousi et al., 2015; Farvin et al., 2016; Bahl et al., 2011; Sung et al., 2006) and performance testing (Garousi and Zhi, 2013; Garousi et al., 2015; Phillips and Alam, 2003) are less prevalent in emerging SE industries (Jahan et al., 2019).	Though Bangladesh has similarities with emerging industries in terms of security testing, 'no security' is less prevalent in Bangladesh.

- *Requirements gathering (Q7):* according to the Figure 1(b), using plain text (44%) and storyboard (41%) are the most widely used requirement-gathering methods. This result is similar to the survey of Vonken et al. (2012). From their study, we can find that the textual description of specifying requirements is the most favorite approach in the Netherlands.
- *Timeline of development activities (Q8):* according to the study of Wang and Galster (2018), during system design and development, most time is spent on implementation and coding, and relatively less time is spent on maintenance in New Zealand similar to Bangladesh, as revealed in our study. On the other hand, requirement analysis requires the second most time in Bangladesh, according to 45% respondents to our survey. In contrast, in Malaysia, as per Baharom et al. (2005), most organisations spend from 5% to 20% of their efforts in planning and requirement analysis. However, if we compare Bangladesh with technologically advanced regions like Japan, India, Europe, etc., with the help of the study of Cusumano et al. (2003), we observe that there exists a substantial difference in the timeline of development activities with Bangladesh. Their study has reported that architectural, functional, and design specification documents are the most used and well-regarded practice in those regions rather than just writing code with

minimal planning and documentation. But in Bangladesh, the implementation phase gets the most priority over other development stages.

RQ2-D1. Software development methodologies used: The agile method is the most popular approach for software development across the globe except in some areas (e.g., New Zealand). Bangladesh SE industry mainly uses plain text like other countries to collect requirements (e.g., Netherlands). In comparison with technologically advanced regions, Bangladesh SE industry lags in giving value to system design and planning.

Table 6 Summary of the comparison of development practices between advanced and emerging countries

Dimension	*Developed*	*Emerging*
D1	Advanced software industries give high importance to planning their products.	In Bangladesh, the implementation phase gets the highest priority. Like Malaysia an ad hoc-based software development practice exists in Bangladesh.
D2	Web as tech platforms and Windows as OS are the dominating platforms in New Zealand. In Turkey, Java and Python are popular programming languages.	Like other countries, web-based platforms have widespread demand in Bangladesh. Bangladesh SE industry has a similarity with Turkey in terms of the popularity of Java and Python.
D3	Unit testing is highly practiced in technologically advanced SE industries. The level of test automation is not satisfactory in the European SE industry.	In Bangladesh, unit testing is less extensive than the advanced SE industry practice. However, it has similarities with the European SE industry in test automation.
D4	Security and performance testing are common practices in advanced industries.	Though Bangladesh has similarities with emerging industries in terms of security testing, 'no security' is less prevalent in Bangladesh.

4.2.2 Software development tools and techniques used (D2)

We compare our observations from three questions: technology platform (Q9), operating system (Q10) and programming language (Q11).

For these criteria, we find relevant information mostly from the study carried out on the SE industry of New Zealand. The comparisons are discussed below.

- *Technology platforms (Q9):* as shown in Figure 3(a), most of our survey respondents (80%) work in web platforms. This outcome is similar to the result of the survey of Wang and Galster (2018) which reveals that most of the developers work in web platforms in New Zealand.
- *Operating systems (Q10):* we have found that Windows is mostly used among developers in New Zealand based on the study (Wang and Galster, 2018) whereas Linux is mostly used by Bangladeshi developers as found in our survey [presented in Figure 3(b)].

- *Programming languages (Q11):* according to the study of Wang and Galster (2018), Java ranks quite low in New Zealand. However, it is the second most used programming language in Bangladesh as per our study reported in Figure 3(c) as well as the most used language in Turkey (Garousi et al., 2015). Again, Python did not have a good standing in the ranking of languages used in New Zealand; on the other hand, it is used significantly in Bangladesh.

RQ2-D2. Software development tools and techniques used: Like other SE industries, the web is the main technology platform in Bangladesh. Linux is the preferred OS in the Bangladesh SE industry, while it is Windows in New Zealand. Although Java and Python are popular languages in the SE industry in Bangladesh, we have noticed that they are not very popular in the New Zealand SE industry.

4.2.3 Software testing and DevOps practices used (D3)

We compare the observations from two questions in our survey: software testing practices (Q14) and level of automated testing (Q15). The observations from three questions [tools used in testing and QA (Q16), continuous deployment tools (Q17) and version control (Q18)] could not be compared, because those were not previously asked in the context of other countries. The comparisons are discussed below:

- *Software testing practices (Q14):* from our study, as per Figure 6(a), we see an interesting point that unit testing (53%) and functional testing (49%) are moderately used in Bangladesh, whereas from Garousi and Zhi (2013) and Wang and Galster (2018), we can see that relatively a high percentage of survey respondents in both Canada and New Zealand reported practice unit testing, i.e., 79.27% and 73%, respectively. On the other hand, the adoption of acceptance testing and UI testing in Bangladesh are quite similar to these countries. In Malaysia, based on Baharom et al. (2005), the authors reported that according to their survey, unit testing (68.29%), integration testing (78.05%), system testing (85.37%) and acceptance testing (78.05%) are used by most organisations in a high percentage, and about half of the organisations are carrying out alpha and beta testing.
- *Level of automated testing (Q15):* we have used the Likert scale to measure the level of automated testing. In the Likert scale, the level is mapped with the usage of automated testing of the respondents. The more the level is the more the respondents use automated testing in their software projects. We have found that as per Figure 6(b), around 25% of our respondents are highly concerned that they have to use automated testing for their projects, while around 35% of our respondents have expressed medium-level concern, and the remaining are hardly concerned about using automated testing. In automated testing practices, we have found that Bangladesh is quite similar to Canada and Turkey, where on average 80% respondents use manual testing and 20% respondents use automated testing (Garousi and Zhi, 2013; Garousi et al., 2015). In New Zealand, though 84% respondents use manual testing, 62% respondents use automated testing (Wang and Galster, 2018).

RQ2-D3. Software testing and DevOps practices used: Though automated testing could indicate rigorous, systematic, and repeatable testing practices, the SE industry of many developed countries as we studied in the literature performs more than 80% of their testing manually. Bangladesh SE industry is following the same pattern. This indicates that test automation still has room for more adoption among the software companies of many developed countries as well as in Bangladesh.

4.2.4 Performance and security measures used (D4)

We compare observations from our three open-ended questions with regard to the security and performance-related features: how do you ensure performance, scalability (Q21, Q22), and security (Q23) in your software products? The comparisons are discussed below:

- *Performance (Q21, Q22):* 21.82% respondents of our survey use performance testing to ensure the performance of their product. However, it is the second least practiced measure among all the measures. Garousi et al. (2015) found that developers mark the lack of performance testing as the main challenge in software maintenance in the Turkish software industry. However, the scenario is different for the Canadian software industry. Participants of the survey of Garousi and Zhi (2013) reported that 40% of them conduct performance testing, and 30% of their total testing effort is spent on performance testing. The New Zealand software industry follows a practice similar to Canada as reported by Phillips and Alam (2003).

  The practice in Bangladesh matches that of Pakistan.

  In the survey of Jahan et al. (2019), only 5% of participants reported conducting performance testing. It seems that performance testing is less popular in growing software industries such as Bangladesh and Pakistan.

  Peer code review is the least practiced measure in the Bangladesh software industry. However, in the Turkish software industry, peer review is ranked as the most frequent activity (Garousi et al., 2015) (ranked five on a five-point Likert scale), though the practice is only limited to code review. Architecture/design review is hardly practiced in Turkey (ranked first on a five-point Likert scale). We found that peer review is limited to only code review in the Bangladeshi software industry, and only 7.27% of our participants reported practicing peer code review.

  There is no study focusing on scalability practices in specific software industries, so it is not easy to compare scalability practices. However, in a study on Finnish DevOps practice, Laihonen (2018) found that the Finnish software industry prefers cloud services as it helps them automate quality assurance. He also reported that DevOps practitioners are inclined towards micro-service architecture rather than monolithic architecture. Hussain et al. (2017) conducted a study to identify trends in the DevOps practices in New Zealand. For this study, besides interviewing the DevOps, they examined the job advertisements for a DevOps role. They found that containerisation technologies (e,g., Docker, Kubernetes) have a high demand in the New Zealand software industry. 94% of job advertisement requires expertise in one or multiple containerisation technologies. This indicates the

popularity of docker technology in the New Zealand software industry. However, in the Bangladesh software industry, the scenario is different. Cloud services are the second most popular (28.85%) measure to ensure scalability where the use of containerisation technologies are not that much popular (3.85%)

- *Security (Q23):* a very few percentages (9.56%) of our survey respondents reported not using any security measures in their product. This practice is also prevalent in the Indian and Malaysian software industries. Bahl et al. (2011) reported that due to misalignment with organisation design, goal, and strategy in some Indian software firms, security measures are not practiced. In a study with Malaysian developers, Farvin et al. (2016) found that 31% of respondents think it is not required to add security in the requirement analysis of a product. Basharat et al. (2013) reported a sense of false security in the small software industry, and standard security practices are hardly followed. It is likely to be applicable to the industry in Bangladesh as well. From the response to a survey on the Turkish software industry, Garousi et al. (2015) ranked different design activities in terms of frequency. Security architecture was ranked second out of five (five is for always-used activities and one is for never-used activities). The ranking shows that security architecture is not a frequent activity in the Turkish software industry. However, in our survey, we see very few respondents reported practicing security design principles while designing system architecture. Our survey found that 5.56% of respondents rely on security architecture/security design principles (application side measures) to ensure the security of their product. The software industries of Bangladesh, Turkey and New Zealand have a resemblance in the practice of security testing. Garousi et al. (2015) reported that security testing is the least widely used among all kinds of testing (e.g., unit testing, integration testing). Sung et al. (2006) found that in the New Zealand software industry, security testing and recovery testing practices are low compared to functional testing. The scenario is the same for Bangladesh; we found that 16.67% of respondents reported security testing to ensure security.

RQ2-D4. Security and performance measures used: Compared to other emerging SE industries, Bangladesh has less practice of security consideration. However, we have observed a lack of testing practices to ensure security and performance. Besides, the Bangladesh SE industry lags in using new technologies (e.g., container, cloud).

5 Implications of findings

The findings from our study can guide the following major stakeholders in SE:

1 SE tool creators to develop a usable and affordable automated testing framework that can be accessible to emerging countries

2 SE researchers to compare and contrast software development practices in emerging countries with respect to region-specific and global trends

3 SE career enthusiasts who would like to participate in the high-growth software industries in emerging countries

4 SE security and performance practitioners to develop techniques to better enforce such crucial non-functional requirements in software products in emerging countries

5 SE industry leaders to offer customised region-specific software products and to promote diversity irrespective of regions.

We discuss the implications below:

5.1 Implication for tool creators

From the response of Q17, we observe that using tools for continuous deployment is commensurate to the years of professional experience. Personnel of senior level are more likely to work in DevOps-related fields. The tool needs for developers may vary depending on their experience types. Indeed, we see clear evidence of the profession types between novice and experienced SE developers in Bangladesh. We see that the percentage of software developers decreased with the increase in professional experience, but the trend is reversed in the case of managers. This is natural as senior professionals usually assume managerial responsibility. From Q7, we see that the employees up to mid-senior level, most of whom are developers, tend to use plain text to gather requirements of a software project, whereas more than 5 years of experience professionals prefer storyboards. For Q8, we see that the implementation among all other development activities is the main concern for all levels of experienced employees. However, the ratio of the top two activities points out that the more senior an employee is, the more he/she tends to analyse the requirements of a software project. As per the Q10, we find that at the initial stage of their career, professionals are inclined to prefer Windows most and then they mostly use Linux in mid-career and gradually they tend to use macOS in late-career. It might indicate that employees were proficient in Windows before the start of their careers. Since the percentage of developers is dominant up to mid-career and the percentage of managers is higher among the late-career professionals, we may deduce that the developers are inclined to use Windows and Linux, and managers prefer macOS for their managerial tasks. Another explanation is that most of the software companies in emerging countries can afford to give expensive mac machines to seniors only. Therefore, tool creators may take into account the above findings during their tool development for diverse SE experience cohorts.

We have also found that rigorous testing practice is not prevalent in Bangladesh. The difference in testing effort between the established software industry (e.g., Canada) and Bangladesh is high (please see RQ2 in Subsection 4.2). The scenario is also true for other developing countries like Pakistan (Jahan et al., 2019). Similar to some other emerging industries such as India (Bahl et al., 2011) and Pakistan (Basharat et al., 2013), security testing is less prioritised in Bangladesh.

While these findings show a difference in the adoption of testing tools and non-functional measures between emerging and developed countries, the reasons could be multi-faceted. For example, one reason for the prevalence of less automated testing in emerging countries like Bangladesh is that most of the developed software products are web-based. This means that developers in Bangladesh need to test their GUI-based software products. Proper software testing tool support for GUI testing is limited in number compared to the testing of source code. Good GUI-based testing tools are also

not free or open-source. This makes it hard for developers in emerging countries to learn and use GUI-based testing frameworks. Since the budget for software development in emerging countries is usually low, they can hardly afford expensive tools. Therefore, while unit testing is practiced widely in Bangladesh like in any other country, the lack of automated tool-based testing is due to the availability of good, usable, and affordable testing tools. Therefore, SE tool creators may offer low-cost services with at least the basic features to address this affordability issue in emerging countries. This is important because a software product developed in an emerging country like Bangladesh is actually consumed mostly in developed countries (e.g., via outsourcing). Therefore, anyone can suffer from any lingering faults in the developed software products.

5.2 Implication for SE researchers

In SE research, we need to be aware of the current trends in software development practices not only to guide our research along the trends but also to ensure that our research contributions are timely and effective to the current needs of the software industry. In research, we strive to achieve generalisability of our findings because otherwise, we run the risk of becoming too niche or specific and may not cater to a global audience. Our study results of software development practices in an emerging country like Bangladesh and the comparison of such practices against multiple countries worldwide show that development practices, tools, and techniques all could vary across the countries. Therefore, it may not be always possible to observe or enforce similar development practices across the globe, even when certain practices may be perceived as superior to others. A major reason for the observed differences between emerging countries and developed countries is that outsourcing is still a major source of revenue for the software industries in emerging countries (LICT, 2019). This explains why product design and overall architecture design are not widely practiced in the software industries of emerging countries.

Another limitation of the SE industry here was the lack of automated testing and security audit tools. Automatic bug detection, program repair, etc., are very popular research topics these days. However, researchers have to struggle to find real-world test opportunities for their developed tools/techniques as companies in developed countries are usually very sensitive to exposing their code to third-party tools. Emerging countries such as Bangladesh can provide good opportunity to R&D efforts provided some assurance is there to be able to use those later at affordable cost.

5.3 Implication for SE career enthusiasts

We have found that certain languages (e.g., Java, JavaScript, etc.) and frameworks (e.g., Spring, Django, ASP.NET) have extensive use in the software industry of Bangladesh. Therefore, career enthusiasts who aspire to a software development profession in emerging countries may focus on mastering such skills. Universities can update their curricula to meet industry demand. The students aspiring to join the software industry must prepare themselves accordingly to be productive quickly. We have also found less automated testing practices in the industry of Bangladesh. This may be related to a lack of exposure to the testing framework. Similar to Hussain et al. (2020), we also suggest including testing-related courses into the curriculum of universities and

introducing relevant assignments to have hands-on experience with automated testing tools right from the student level. The students in their academic projects should also use state-of-the-art DevOps tools and technologies (e.g., Docker, Kubernetes, Jenkins, etc.) to have good placement in the industry.

Career enthusiasts in emerging countries like Bangladesh can also benefit from the insights we gained by analysing the development practices based on different SE development roles in the industry. Agile is the most practiced requirement-gathering method. The popularity of the agile method is consistent across the different reported roles in our surveys. The second popular method is scrum. In Q8, participants were asked to identify the SDLC activity, where most of the time is spent. Generally, it is expected that participants in the senior role (e.g., manager, team lead) will spend time in requirement analysis and documentation whereas participants in the junior role (e.g., developers, R&D engineer) will spend time in implementation and testing. However, Q8's responses do not match our expectations. Across all roles, implementation is the most time-consuming activity in SDLC. However, testing is not considered one of the most time-consuming activities. Rigorous testing practices may not be prevalent in the Bangladesh SE industry. In most of the roles, Java is the most used language (Java is the second most used language in cases where JavaScript is the most used language). Even in the data engineer role, Java is the most used language, though Python is mostly used in data processing. Like Java, Spring (one framework of Java) is the most widely used framework regardless of role. Java and JavaScript are the most used languages regardless of role. We observed from Q15 and Q13 that developers practice the highest level of automated testing, and they mostly practice unit tests. One of the reasons for the high level of automated testing among developers may be that it is easier to achieve automated testing in unit testing due to different frameworks/libraries. Affordable user-friendly automated test tools offering libraries/services for other types such as functional, integration/API, performance, and security testing can encourage higher automated testing levels for other roles. Developers mostly practice the highest level of automated testing. Affordable user-friendly automated testing frameworks for other types of testing may increase the practice of automated testing.

Findings like the above can guide a novice developer in an emerging country to decide how to shape his/her career progression in such a diverse and evolving software ecosystem.

5.4 Implication for software security and performance practitioners

More than 9.5% of our respondents responded that they do not take any measures to mitigate the security risk. The common reason for not taking any measures is:

1 the product is an early stage

2 the respondents' role does not require any product security measures.

The reason for not taking any security measures is different from North America (Assal and Chiasson, 2019). The reasons are:

1 there are no formal test plans,

2 lack of knowledge regarding testing tools.

Though respondents do not think about product security initially, it is recommended (Chandra et al., 2009; Azham et al., 2011) to plan security tests and product security from the design phase.

Modern frameworks provide the basic security of the solution. Moreover, some framework provides enhanced, focused, customised security through a plug-in or add-on, e.g., Spring security, and Spring-cloud security. Framework-based security is a growing practice in the software industry (Alssir and Ahmed, 2012). The practice in the Bangladesh SE industry matches the global practice. According to our survey, it is the second most popular measure to mitigate security risk. Survey respondents have reported using OWASP, HDIV, and Spring security. Srinivasan and Sangwan (2017) have conducted a comparison among the popular web frameworks based on security. Based on five criteria, they ranked the frameworks, and all of the mentioned frameworks of our respondents are in the top 10 list. It seems that secure software engineering practice is prevalent in the Bangladesh SE industry.

In a survey of 237 software professionals, Elahi et al. (2011) found that 51% of respondents maintain at least one security standard, and 19% of respondents maintain ISO 17799 security standard. On the contrary, about 29.63% of respondents in the software industry in Bangladesh do not maintain any security standards. It is clear that security standards are not that much practiced in this SE industry.

According to Smith and Williams (2003), efficient architecture and continuous monitoring tools are two of the twenty-four best practices to ensure software performance. The respondents report both practices. However, Smith et al. presented twenty-one other best practices, and we have not found other practices in our survey. It is clear the SE industry of Bangladesh only practices a few best measures for ensuring software performance.

Bondi (2000) have listed four scalability types to ensure software capability to scale; however, we observe only one scalability type (load scalability) in our responses. To ensure software scalability, the use of cloud services is one of the most popular strategies (Gao et al., 2011) in recent times. Another popular strategy is the use of microservices. Microservices and cloud services together allow the user to scale up and down any system dynamically. Cáceres et al. (2010) reported that cloud services and microservices-based architecture are generally used together to ensure scalability. In the Bangladesh SE industry, this practice may be prevalent. The use of cloud service and efficient use of architecture are the second and third most popular topics among those received from our respondents.

Though cloud-based service is used by software companies, container-based tools like Docker and Kubernetes are not sufficiently used compared to the developed countries as evident from the survey response. Those who are interested in DevOps engineering should extensively learn these technologies to ensure efficient container orchestration and deployment that scales.

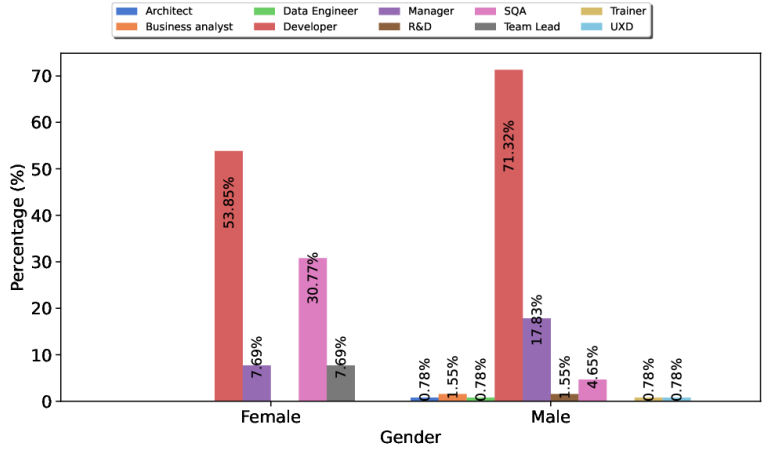
Overall, the findings show that there is much scope for improvement to ensure proper security and performance measures in the software development practices of emerging countries. Multi-faceted efforts are warranted in the development of affordable tools and techniques for emerging countries, enforcement of widely accepted and measurable industry standards across the regions, and the proper training of the security and performance principles to the developers in emerging countries.

5.5 Implication for industry leaders

This study found that the Bangladesh software industry lags in adopting some of the current industry trends such as in the use of automated testing and security analysis tools, DevOps technologies, etc. Whereas user-friendly professional tools are very expensive, there are open-source alternatives that demand significant learning effort. Since most of the companies of the emerging SE industry of Bangladesh cannot afford professional tools and also have a scarcity of experienced resources (as evident from the experience report of survey respondents), industry leaders have to take the book initiative. One step may be to arrange special training on difficult-to-learn tools such as Kubernetes or open-source security testing tools with funding from government or international donor agencies. Another initiative can be promoting specialised companies to offer service to the industry in the demanding areas of security testing or DevOps engineering.

We have identified that the participation of females in the Bangladesh SE industry is significantly less than that of male participants, a global problem across countries that needs attention and measures by indutry leaders to properly address. In our survey, 90.1% of participants were male and 9.9% participants were female, which is slightly better than the Stack Overflow (SO) survey (Stack Overflow, 2017, 2018, 2019, 2020) (in Stack Overflow 8% respondents marked them as female). It is often said that females are less represented in STEM. As the SE industry is directly related to STEM, the claim may be true in the SE industry. The proportion of male and female participants in our survey supports the claim of under-representation. To get an overview of the Bangladesh SE industry by gender in Figure 9, we plotted the participants' roles grouped by gender.

Figure 9 Gender-based role (see online version for colours)



In terms of roles, the proportion of developers among female participants is comparatively lower than that of male participants. However, the scenario is the opposite for the manager and team lead role. Also, among ten different roles, female participants hold only four types of roles. Our findings align with the survey result of Hussain et al. (2020). However, the result of the SO survey (Stack Overflow, 2020) is slightly

different. In the SO survey, female respondents work mostly as data scientists, business analysts, QA, and developers.

James et al.'s (2017) survey on male and female software professionals found that men are more likely to be in senior positions than women. We also observed a similar scenario. However, in our case, the observation is not statistically significant ($p = 0.27$ based on the Mann-Whitney U test). Similar to us, James et al. (2017) also found that male software practitioners tend to be older than female practitioners and female practitioners tend to leave jobs in mid-career.

Previously, Hussain et al. (2020) expressed a concern that there may be bias in the hiring process of the industry. Overall, the under-representation of females and minorities in software industries worldwide is a prevalent and ongoing concern. This was also reported in the 2020 Stack Overflow developer survey, where more than 90% respondents were males. Our findings confirm similar trends in Bangladesh. Therefore, proper measures need to be taken to encourage equity, diversity, and inclusion in software industries across the regions.

6 Threats to validity

We discuss the threats to the validity of our studies following common guidelines for empirical studies (Wohlin et al., 2000).

- *Construct validity* is mainly concerned with the extent to which the study objectives truly represent the theory behind the study (Wohlin et al., 2012). In our study, we have used an open coding strategy to label the survey responses. The nature of this coding strategy may introduce researcher bias into coded labels. To mitigate the issue, the labels have been coded by two individuals, and the codes are accepted when there is a reasonable agreement among the coders. Another issue can be whether our data actually represents real-world SE practices. This study counted the votes and made statistical inferences, which is common in survey-based studies. It is believed that voting data can, to a certain extent, reflect the opinions of the majority. It was previously observed (Garousi et al., 2015) that people tend to form their answers close to expected answers when evaluated. To mitigate the threat, before the survey, we informed participants that our motive in this survey was to get a decent understanding of current practices, and we do not intend to collect any personally identifiable data. Construct threats may also be introduced by a misleading interpretation of the survey questions. We conducted a preliminary survey and interview session with some participants to rule out any ambiguity from survey questions and thus tried to reduce such risk.
- *Internal validity* is a property of scientific studies that refers to how well a study has been conducted. A threat to internal validity in this study is inherent in the participant selection bias. We used several social platforms and personal connections to reach as many participants as possible. Another threat could arise from the placement of the options in a multiple-choice question. It is often observed that survey participants often show bias towards the first option in any multiple-choice question (Uddin et al., 2019). However, in one of the multiple-choice questions (Q9), the 'Web' option was placed at the bottom of the list. Despite this placement, we observed most of the participants selected the web

as a technology platform. Moreover, from the personal practical experience of the authors, there is no bias in this opinion. In terms of reliability, there were no criteria for selecting participants. We have recruited participants who have worked in the emerging software industry. Moreover, our study represents participants from different levels of the hierarchy. Thus, replicating this study in the context of another emerging country should result in similar findings. Moreover, we have shared the replication package of this study at https://git.io/JLtxI which can be used to replicate the study in other contexts.

- *External validity* is concerned with the generalisation of the study result. In Bangladesh, there are around 1,100 software companies and around 3,00,000 IT professionals across diverse domains (e.g., outsourcing, consulting, etc.) (Basis, 2018). In our study, we have participants from almost all the groups of the Bangladesh software industry. While more responses would have offered more proof of generalisability, we note that we already observed saturation in our manual coding of themes and labels (i.e., during open coding). We also found a considerable concentration of professionals supporting each development aspect we studied in the paper (e.g., testing practices, etc.). As we noted in Subsection 4.2, we observed certain similarities between our findings in Bangladesh against other emerging countries like Malaysia. We also found some commonalities and differences in the development practices between the emerging countries and the developed countries (please see Subsection 4.2).

7 Conclusions

This study identifies the general practices, obstacles, and limitations of the software development industry in Bangladesh which represents an emerging software industry. To reach this goal, we conducted a series of interviews to design survey questions related to software development practices and then analysed the survey responses to get our research questions. It was revealed that automated testing and deployment practices are quite low compared to the established software industry. Although we have found that software security consciousness is higher than in other emerging software industries, security testing practices and standards lag behind developed countries. We have noticed the low participation of women in the software industry. Further research may be conducted to determine any biases or barriers for women in the SE industry. These findings can help developers, industry owners, and academia work to improve the industry from their own positions and perspectives.

References

Almomani, M.A.T., Basri, S., Mahmood, A.K.B. and Bajeh, A.O. (2015) 'Software development practices and problems in Malaysian small and medium software enterprises: a pilot study', *2015 5th International Conference on IT Convergence and Security (ICITCS)*, IEEE.

Alssir, F.T. and Ahmed, M. (2012) 'Web security testing approaches: comparison framework', *Advances in Intelligent and Soft Computing*, pp.163–169, Springer, Berlin, Heidelberg.

AlSubaihin, A., Sarro, F., Black, S., Capra, L. and Harman, M. (2019) 'App store effects on software engineering practices', *IEEE Transactions on Software Engineering*, Vol. 47, No. 2, p.1.

Assal, H. and Chiasson, S. (2019) 'Think secure from the beginning', *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems – CHI '19*, ACM Press.

Azham, Z., Ghani, I. and Ithnin, N. (2011) 'Security backlog in scrum security practices', *2011 Malaysian Conference in Software Engineering*, IEEE.

Baharom, F., Deraman, A. and Hamdan, A.R. (2005) 'A survey on the current practices of software development process in Malaysia', *Journal of Information and Communication Technology (JICT)*, Vol. 4, No. 1, pp.57–76.

Bahl, S., Wali, O.P. and Kumaraguru, P. (2011) 'Information security practices followed in the Indian software services industry: an exploratory study', *2011 Second Worldwide Cybersecurity Summit (WCS)*, pp.1–7.

Basharat, I., Fatima, M. Nisa, R., Hashim, R. and Khanum, A. (2013) 'Requirements engineering practices in small and medium software companies: an empirical study', in *Science and Information Conference (SAI)*, IEEE, pp.218–222.

Basis (2018) *IT and ITES Industry Overview*, Bangladesh Association of Software and Information Services [online] https://basis.org.bd/public/files/publication/5e127d7cb6967ba96136a3b168568073f9800e5b0f5b9.pdf (accessed 1 February 2022).

Begum, Z., Khan, M.S.A., Hafiz, M., Islam, M.S. and Shoyaib, M. (2009) 'Sofware development standard and software engineering practice: a case study of Bangladesh', *Journal of Bangladesh Academy of Sciences*, Vol. 32, No. 2, pp.131–139.

Bhuiyan, S.M.A.R., Rahim, M.S. Chowdhury, A.Z.M.E. and Hasan, M.H. (2018) 'A survey of software quality assurance and testing practices and challenges in Bangladesh', *International Journal of Computer Applications*, Vol. 180, No. 39, pp.1–8.

Bondi, A.B. (2000) 'Characteristics of scalability and their impact on performance', *Proceedings of the second international workshop on Software and performance – WOSP '00*, ACM Press.

Cáceres, J., Vaquero, L.M., Rodero-Merino, L., Polo, Á. and Hierro, J.J. (2010) 'Service scalability over the cloud', *Handbook of Cloud Computing*, pp.357–377, Springer, USA.

Chandra, S., Khan, R.A. and Agrawal, A. (2009) 'Software security factors in design phase', *Information Systems, Technology and Management*, pp.339–340, Springer, Berlin, Heidelberg.

Cohen, J. (1960) 'A coefficient of agreement for nominal scales', *Educational and Psychological Measurement*, Vol. 20, No. 1, pp.37–46.

Creswell, J.W. (2013) *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 4th ed., Sage Publications, Inc.

Cusumano, M., MacCormack, A., Kemerer, C. and Crandall, B. (2003) 'Software development worldwide: the state of the practice', *IEEE Software*, Vol. 20, No. 6, pp.28–34.

Dutta, S., Lee, M. and Wassenhove, L.V. (1999) 'Software engineering in Europe: a study of best practices', *IEEE Software*, Vol. 16, No. 3, pp.82–90.

Elahi, G., Yu, E., Li, T. and Liu, L. (2011) 'Security requirements engineering in the wild: a survey of common practices', *2011 IEEE 35th Annual Computer Software and Applications Conference*, IEEE.

Farvin, S., Baharom, F., Deraman, A., Yahaya, J. and Haslina, H. (2016) 'An exploratory study on secure software practices among software practitioners in Malaysia', *Journal of Telecommunication, Electronic and Computer Engineering*, Vol. 8, No. 8, pp.39–45.

Freelon, D. (2020) *ReCal2: Reliability for 2 Coders* [online] http://dfreelon.org/utils/recalfront/recal2/.

Gao, J., Pattabhiraman, P., Bai, X. and Tsai, W.T. (2011) 'SaaS performance and scalability evaluation in clouds', *Proceedings of 2011 IEEE 6th International Symposium on Service Oriented System (SOSE)*, IEEE.

Garousi, V., Coşkunçay, A., Betin-Can, A. and Demirörs, O. (2015) 'A survey of software engineering practices in turkey', *The Journal of Systems and Software*, Vol. 108, No. 10, pp.148–177.

Garousi, V. and Zhi, J. (2013) 'A survey of software testing practices in Canada', *Journal of Systems and Software*, Vol. 86, No. 5, pp.1354–1376.

Github (2020) *The State of the Octoverse* [online] https://octoverse.github.com/ (accessed 1 February 2022).

Groves, L., Nickson, R., Reeve, G., Reeves, S. and Utting, M. (2000) 'A survey of software development practices in the new zealand software industry', *Proceedings of Australian Software Engineering Conference (ASWEC)*.

Hussain, I., Hasan, A.S., Yasir, R.M., Kabir, A. and Ahmed, S.I. (2020) 'How well does undergraduate education prepare software engineers? Perspectives of practitioners in Bangladesh', *2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T)*, IEEE.

Hussain, W., Clear, T. and MacDonell, S. (2017) 'Emerging trends for global DevOps: a New Zealand perspective', *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*, IEEE.

Jahan, M.S., Riaz, M.T., Kashif, and Abbas, M. (2019) 'Software testing practices in IT industry of Pakistan', *Proceedings of the 6th Conference on the Engineering of Computer Based Systems – ECBS '19*, ACM Press.

James, T., Galster, M., Blincoe, K. and Miller, G. (2017) 'What is the perception of female and male software professionals on performance, team dynamics and job satisfaction? insights from the trenches', *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*, IEEE.

JetBrains (2019) *The State of Developer Ecosystem 2019* [online] https://www.jetbrains.com/lp/devecosystem-2019/ (accessed 1 February 2022).

JetBrains (2020) *The State of Developer Ecosystem 2020* [online] https://www.jetbrains.com/lp/devecosystem-2020/ (accessed 1 February 2022).

Joyce, M. (2013) *Picking the Best Intercoder Reliability Statistic for Your Digital Activism Content Analysis*.

Klotins, E., Unterkalmsteiner, M. and Gorschek, T. (2018) 'Software engineering in start-up companies: an analysis of 88 experience reports', *Empirical Software Engineering*, Vol. 24, No. 1, pp.68–102.

Krippendorff, K. (2004) 'Reliability in content analysis: some common misconceptions and recommendations', *Human Communication Research*, Vol. 30, No. 3, pp.411–433.

Laihonen, P. (2018) *Adoption of Devops Practices in the Finnish Software Industry: An Empirical Study*, Master's thesis.

Landis, J.R. and Koch, G.G. (1977) 'The measurement of observer agreement for categorical data', *Biometrics*, Vol. 33, No. 1, p.159.

LICT (2019) *IT and ITES Industry Statistics of Bangladesh*, Bangladesh Computer Council (BCC) [online] https://lict.gov.bd/uploads/file/strategic/strategic_5e4cb5ed90760.pdf (accessed 1 February 2022).

Lu, C-J. and Shulman, S.W. (2008) 'Rigor and flexibility in computer-based qualitative research: introducing the coding analysis toolkit', *International Journal of Multiple Research Approaches*, Vol. 2, No. 1, pp.105–117.

Phillips, C. and Alam, J. (2003) 'Software engineering practices and tool support: an exploratory study in New Zealand', *Australasian Journal of Information Systems*, Vol. 11, No. 1, pp.37–54.

Rahim, M.S., Hasana, M.H., Chowdhury, A.E. and Das, S. (2017) 'Software engineering practices and challenges in Bangladesh: a preliminary survey', *Journal of Telecommunication*, Vol. 9, No. 3, pp.163–169.

Scott, W.A. (1955) 'Reliability of content analysis: the case of nominal scale coding', *The Public Opinion Quarterly*, Vol. 19, No. 3, pp.321–325.

Seabold, S. and Perktold, J. (2010) 'Statsmodels: econometric and statistical modeling with Python', *9th Python in Science Conference*.

Sison, R., Jarzabek, S., Hock, O., Rivepiboon, W. and Hai, N. (2006) 'Software practices in five asean countries: an exploratory study', *Proceeding of the 28th International Conference on Software Engineering (ICSE)*.

Smith, C.U. and Williams, L.G. (2003) 'Best practices for software performance engineering', *29th International Computer Measurement Group Conference, Proceedings*, 7–12 December, Computer Measurement Group, Dallas, Texas, USA, pp.83–92.

Srinivasan, S.M. and Sangwan, R.S. (2017) 'Web app security: a comparison and categorization of testing frameworks', *IEEE Software*, Vol. 34, No. 1, pp.99–102.

Stack Overflow (2017) *Stack Overflow Survey 2017* [online] https://insights.stackoverflow.com/survey/2017 (accessed 1 February 2022).

Stack Overflow (2018) *Stack Overflow Survey 2018* [online] https://insights.stackoverflow.com/survey/2018 (accessed 1 February 2022).

Stack Overflow (2019) *Stack Overflow Survey 2019* [online] https://insights.stackoverflow.com/survey/2019 (accessed 1 February 2022).

Stack Overflow (2020) *Stack Overflow Survey 2020* [online] https://insights.stackoverflow.com/survey/2020 (accessed 1 February 2022).

Sung, P., Sung, B. and Paynter, J. (2006) 'Software testing practices in New Zealand', *19th Annual Conference of the National Advisory Committee on Computing Qualifications (NACCQ 2006), Proceedings*, Wellington, New Zealand, pp.273–282.

Uddin, G., Baysal, O., Guerrouj, L. and Khomh, F. (2019) 'Understanding how and why developers seek and analyze API-related opinions', *IEEE Transactions on Software Engineering*, Vol. 47, No. 4, p.1.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P. and SciPy 1.0 Contributors (2020) 'SciPy 1.0: fundamental algorithms for scientific computing in Python', *Nature Methods*, Vol. 17, No. 2, pp.261–272.

Vogt, W. and Johnson, R.B. (2005) *Dictionary of Statistics and Methodology – A Non-technical Guide for the Social Sciences*, 3th ed., Sage Publications, Inc.

Vonken, F., Brunekreef, J., Zaidman, A. and Peeters, F. (2012) *Software Engineering in the Netherlands: The State of the Practice*, Technical Report Series, Delft University of Technology, Software Engineering Research Group.

Wang, D. and Galster, M. (2018) 'Development processes and practices in a small but growing software industry – a practitioner survey in New Zealand', *Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) (ESEM'18)*, 11–12 October, ACM Press, Oulu, Finland, pp.1–10.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B. and Wesslén, A. (2000) *Experimentation in Software Engineering: An Introduction*, Kluwer Academic Publishers, Norwell, MA, USA.

Wohlin, C., Runeson, P., Hst, M., Ohlsson, M.C., Regnell, B. and Wessln, A. (2012) *Experimentation in Software Engineering*, Springer Publishing Company, Incorporated.

Zafar, I., Shaheen, A., Nazir, A.K., Maqbool, B., Butt, W.H. and Zeb, J. (2018) 'Why Pakistani software companies don't use best practices for requirement engineering processes', *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, IEEE.