Turkish Journal of Electrical Engineering and Computer Sciences

Volume 28 | Number 4 Article 3

1-1-2020

Assessment of environmental factors affecting software reliability: a survey study

ALPER ÖZCAN

ÇAĞATAY ÇATAL

CENGİZ TOĞAY

BEDİR TEKİNERDOĞAN

EMRAH DÖNMEZ

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2020) 28: 1841 – 1858

doi:10.3906/elk-1907-49

Assessment of environmental factors affecting software reliability: a survey study

Alper ÖZCAN[1], Çağatay ÇATAL[2,*], Cengiz TOGAY[3], Bedir TEKİNERDOĞAN[4], Emrah DÖNMEZ[5]

[1]Research & Development Group, Softtech Research Center, İstanbul, Turkey
[2]Department of Computer Engineering, Bahçeşehir University, İstanbul, Turkey
[3]Department of Computer Engineering, Uludağ University, Bursa, Turkey
[4]Information Technology Group, Wageningen University & Research, Wageningen, Netherlands
[5]Department of Computer Engineering, İnönü University, Malatya, Turkey

Received: 08.07.2019 • **Accepted/Published Online:** 21.02.2020 • **Final Version:** 29.07.2020

Abstract: Currently, many systems depend on software, and software reliability as such has become one of the key challenges. Several studies have been carried out that focus on the impact of external environmental factors that impact software reliability. These studies, however, were all carried out in the same geographical context. Given the rapid developments in software engineering, this study aims to identify and reinvestigate the environmental factors that impact software reliability by also considering a different context. The environmental factors that have an impact on software reliability as reported in earlier studies have been analyzed and synthesized. Subsequently, a survey study is conducted to analyze the impact of 32 environmental factors from the perspective of multiple stakeholders. Several statistical analysis methods were applied for the analysis. Data were collected from 24 organizations and 70 software professionals. Most factors shown in top 10 lists of previous studies remain in the top 10 in our study, but their order is different. Testing coverage is now the most significant factor and testing effort is considered as the second most significant factor. The environmental factors defined previously retain their impact. The ordering of the importance of the environmental factors has changed though.

Key words: Software reliability, survey, software engineering, environmental factors

1. Introduction

Software-intensive systems are getting more and more complex with the introduction of new technologies (e.g., blockchain and edge computing), the higher expectations of end users, and the increasing number of development teams and locations necessary for the development of a complex software product (i.e. global software engineering). Codebases in many large-scale projects have now reached hundreds of millions of lines of code with a human intractable number of states. Software systems are nowadays among the most complex entities humankind has ever built [1]. Due to this increasing complexity, software reliability is now one of the most important challenges of software development and hence building a reliable software system requires new tools, methods, and techniques. Although software reliability modeling has been studied since the 1970s and many models have been published so far [2], reliability in software engineering is still an active research area [3–6].

Many different definitions of reliability have been provided, often including the probability of failure-free operation in the hardware domain or continuity of proper service. One of the standard definitions is

*Correspondence: cagataycatal@gmail.com

provided by ISO/IEC SQuaRE [7], which defines software product reliability as the degree to which a software product performs its functions for a certain period under predetermined conditions. In the SQuaRE quality model [7], reliability is a key quality factor that further includes the subcharacteristics of availability, maturity, fault tolerance, and recoverability. Many studies have focused on providing techniques and tools for these characteristics with the aim of building a reliable system.

Despite the broad interest in software reliability analysis, it appears that the majority of the current studies focused on the internal quality of the system and did not explicitly consider the environmental factors that impact the reliability of software systems during the development, testing, and operation [8]. Software development is an inherently complex human activity that is performed in different contexts. A comprehensive reliability assessment requires the consideration of both the internal and environmental factors. While there is no general definition of environmental factors that affect the software reliability [9], there are several studies that explain and investigate these environmental factors [10].

Initially, Zhang and Pham [10] presented 32 environmental factors (e.g., requirements analysis, testing effort, domain knowledge, documentation, human nature, and processors) including each phase of the software development process, teamwork, human nature, and interactions with hardware systems [8] and investigated their rankings and correlations for the software reliability assessment by using a survey among many experts from industry. Later, Zhang et al. [11] performed further analysis to reduce the number of factors with the help of factor analysis and analyzed the relationships between several factors. Fifteen years later, Zhu et al. [8] revisited these 32 factors and analyzed their impact on software reliability using a survey. They compared their findings with those two papers previously published. Since these three studies focused on the software development process of single-release software, later on, they performed another survey study to investigate the impact of environmental factors for multi-release software [9].

All of the studies that we mentioned were performed with experts working in US companies but nowadays globally distributed projects are very common in the software industry and hence we need to gather other perspectives from software experts in other countries. Thus, the present study aims to reinvestigate the impact level of environmental factors in a geographical context different than the US. For this, we chose Turkey, which is one of the top 30 countries for offshore IT services according to Gartner reports [12]. The present study further aims to compare the results with those of earlier studies and reports on the differences and the key lessons learned.

This investigation is performed using a survey study with 70 software experts in the Turkish software industry fulfilling different roles including software managers, software engineers, architects, project managers, analysts, and test engineers. These software experts were asked to evaluate the effect of environmental factors on the software reliability assessment. After the survey results were collected, they were processed with several statistical techniques such as relative weighted method, principal component analysis (PCA), analysis of variance (ANOVA), and correlation analysis and compared with those of previous studies. The survey protocol representing the main steps of the survey study is presented in Section 3. The remaining sections are organized as follows: Section 2 presents the background and related work. Section 3 describes the adopted research methodology of the present study. Section 4 presents the results of the survey study. Section 5 presents the discussion, including the lessons learned and the potential validity threats. Finally, section 6 shows the conclusion and future work.

2. Related work

Software reliability is one of the eight quality characteristics based on ISO/IEC SQuaRE standard [7], which is affected by various different factors. The type and the number of these factors can vary depending on the number of parameters in the analyzed environment. Several studies have been presented in the literature that aim to define and analyze these factors. Their specific goal is first to define the factors affecting the reliability and then analyze the correlation between these factors by using the data acquired from the software organizations.

Zhang and Pham [10] introduced 32 environmental factors to determine the effect of these factors affecting the reliability of software during the software development process. These environmental factors were grouped into five parts: general, analysis & design, coding, testing, and hardware systems. A survey was conducted with software managers and developers in 13 companies such as Chrysler and AT&T that had software projects. They emphasized the significance of factors in development phases and examined the factor correlation. They identified the top five factors as follows: program complexity, programmer skills, testing coverage, testing effort, and testing environment.

Zhang et al. [11] performed an exploratory analysis to investigate the relations between previously determined environmental factors in their study [10]. They aimed to reduce the factor dimension space by combining these factors, analyzed whether the development phases have similar impact on reliability assessment or not, and investigated if there is a correlation between the background information of the participants, such as experience, and their opinions. The top 11 environmental factors were applied for the factor analysis. Four common categories were determined as follows: general factors that were represented under the overall factor, testing efficiency, requirements & specification, and program & skill level. They reported that four software development phases (testing, coding, general, and analysis & design) were equally important for software reliability assessment.

Zhu et al. [8] aimed to reinvestigate the 32 environmental factors in 2015, which were introduced by Zhang and Pham [10] in 2000 because software development has changed dramatically during the intervening 15 years. The survey was performed among software practitioners from 20 organizations. They compared their results with those of the two studies by Zhang et al. [10, 11] and determined the most important factors based on the principle component analysis and the factor ranking methods. Most of the factors in the top 10 list of Zhang et al. [10, 11] were again in the top 10 list in their new study, but there were some changes in their importance order. The top environmental factor was the frequency of program specification change in their study, while it was program complexity in the previous study [10]. They compared the principle components determined in their study with the common factors reported in Zhang and Pham's study [11] and reported that these components are slightly different compared to the previous findings. Zhang and Pham [11] had stated that all phases of software development have equal importance for reliability assessment, but in this study they showed that the testing phase has the highest impact on reliability and analysis & design ranked second. Loganathan and Muthuraj [13] aimed to develop a methodology to decrease the data volume in reliability studies of software. They focused on 34 potential environmental factors important for software reliability. A survey was conducted among 25 software developers selected randomly. An agglomerative hierarchical clustering method was implemented and seven clusters were obtained.

Zhu and Pham [9] recently analyzed the environmental factors on reliability for multirelease software and compared the significant factors of single-release software with those of multirelease software. Since lean software development and agile software methodologies have become dominant approaches for software development in the last decade, multirelease software is more common compared to single-release software. They reported that

60% of factors listed as top 10 factors in previous studies are still in the top 10 list, but the significance order has changed dramatically.

Table 1 presents a summary of the previous studies based on several characteristics. The numbers of participants vary between 23 and 45. There are mostly 32 environmental factors. Relative weighted and analysis of variance methods are the preferred methods. A Likert scale was utilized to score each factor in these studies. SNK and Tukey grouping methods were used to classify factors. Correlation analysis was applied to determine the factor relationships and PCA was used to identify the new principles. The statistical methods are mostly used in these studies. Furthermore, the environmental factors should be investigated with more data to improve the external validity. In addition, there may be some factors that were not introduced so far. These additional factors may be about the nature of the software development, cultural aspects, sector, location, and organizational habits. Demographic data are also important in these studies. Our study is different than the studies explained in the table because most of these surveys were performed among software developers in the USA. Due to the increasing level of global software engineering practices, it is crucial to perform a survey with software practitioners in other countries. Compared to the other studies, we gathered more data (70 participants) and analyzed them to investigate our findings in terms of those in the other studies. Our study presents the perspectives of software practitioners in a country other than the USA for the first time in the literature.

Table 1. Summary of the related work.

	Zhang and Pham (2000)	Zhang et al. (2001)	Loganathan and Muthuraj (2013)	Zhu et al. (2015)	Zhu and Pham (2017)
Data	23 surveys (13 different organization)	35 surveys (13 different companies)	25 different software developers	35 surveys (20 different organizations)	45 surveys (different organizations)
Factors	32	32	34	32	32
Method	Relative weighted method and ANOVA	Factor analysis with Varimax rotation and ANOVA	Clustering the factors and single linkage nearest neighbor	Relative weighted method and ANOVA	Relative weighted method, lasso regression, and stepwise backward elimination
Measurement scale	Likert (1–7)	Likert (1–7)	Special (0–7)	Special (0–7)	Likert (1–7)
Factor ranking	Normalized priority results	Normalized data with relative weighted method	No	Normalized data with relative weighted method	Normalized data with relative weighted method
Factor classification	SNK grouping method according to average values	SNK grouping method and regression analysis	No factor clustering and dendrogram	Tukey grouping method according to average values	Tukey method and multiple linear regression
Correlation analysis	Yes (dependency between factors)	No	No	Yes (dependency between factors)	Yes (dependency between factors)
Dimensionality reduction	No	Regression analysis	PCA	PCA	PCA
Development phases grouping	No	Yes	No	Yes	Yes

3. Methodology

3.1. Survey protocol

In this section we first present the survey design that is used to answer the research question. Initially, the survey design must be able to match the defined objectives and the extracted survey data and analysis must be able to answer the posed research question. Surveys might be designed for different purposes and can be carried out in different ways. The survey design protocol that we describe is based on the protocol as defined by Kitchenham and Phleeger [14–16]. There are several advantages of survey research [17]. First, the empirical data are produced based on real-world observations. Second, it can be generalized to a population and it is more effective in terms of breadth of coverage. Last, the production of the survey data is relatively low cost and can be prepared in a short time. In our case we focus on so-called descriptive survey design, in which it is aimed to capture and describe the current state. In the context of the present study we thus aim to provide a survey design to describe the impact of environmental factors on software reliability in different geographical locations. Descriptive, observational survey designs can be further categorized into cross-sectional, cohort, and case control [18]. Our study can be primarily characterized as a case control study in which the study is retrospective, asking participants about their previous circumstances to help explain a current phenomenon. In particular, we ask stakeholders in software projects about their experiences in the factors that had an impact on software reliability. The survey protocol that we adopt is shown in Figure . Each of these steps in this figure is essential in survey research.

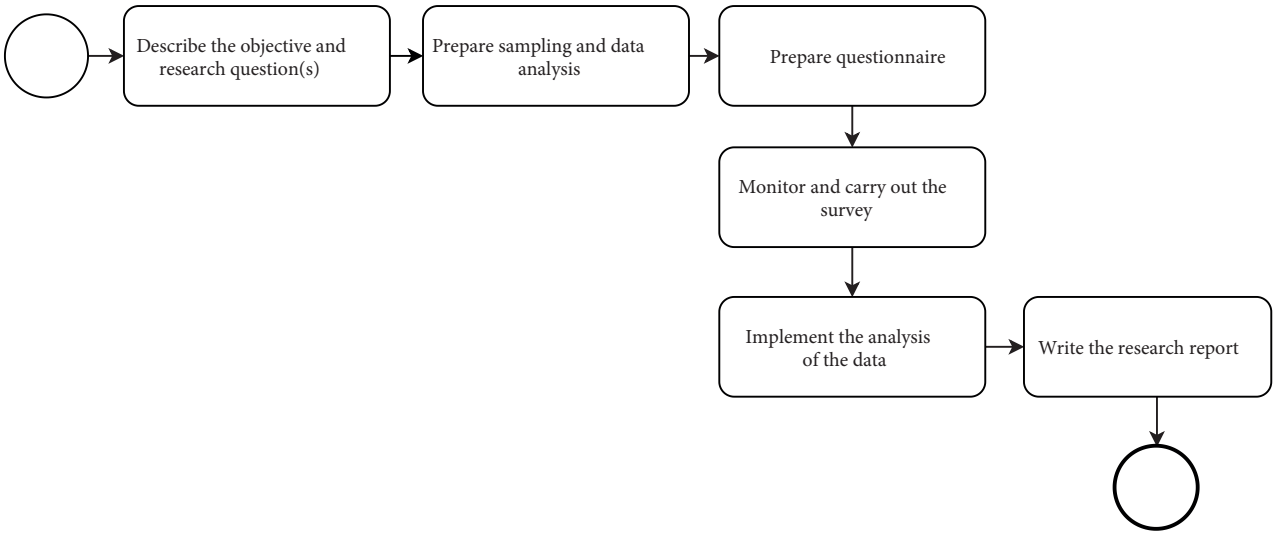

Figure . Survey protocol.

The first step is the description of the objective and the research question(s). A good research study should address a clear research question [17]. The next step is to select a sample of the population because it is impractical to gather data from every individual in the population [19]. Random sampling and nonrandom sampling are the two main categories of sampling. Subsequently, the questionnaire must be prepared and presented clearly. There are several good practices while designing questions for the questionnaire. For instance, the categorization of questions based on subject, numbering the questions, avoiding the use of capital letters only, providing clear instructions, and avoiding the use of two questions in one question (a.k.a., double barreled

question) and questions that have double negatives are some of the good practices that should be taken into account during this step [17]. After the questionnaire is prepared, the survey is mostly performed with a pilot sample of individuals in the population to make sure that the participants understand the questions, all the possible response categories are defined, and no question is systematically missed [17].

In our survey protocol in Figure , this piloting step is considered a part of the Monitor and Carry Out the Survey step and therefore it is not explicitly depicted. While carrying out the survey, a covering letter must be provided to participants to explain the organizations involved in this study, the contact address of the researchers, the purpose of the study, and details of why the participant was selected and how the information will be processed [17]. For implementing the analysis of the data, a considerable amount of time should be planned and spent [17]. Methods that are applied during the analysis depend on the survey design. In the present study, we used relative weighted method (section 3.2.1), PCA (section 3.2.2), ANOVA (section 3.2.3), and correlation analysis (section 3.2.4) as the main approaches for the analysis of the data. The last step of the survey protocol is to write the research report based on the observed results.

3.2. Survey design, execution, and data collection

Replication studies in software engineering play a critical role in building scientific knowledge and, as such, previous experiments and surveys are mostly replicated in different geographical contexts, at different times, and with different kinds of stakeholders. In the present study, we replicated the survey study by Zhu et al. [8] in a different geographical context. We used the same set of questions that were used in both Zhu et al.'s study [8] and Zhang and Pham's study [10] because we aimed to compare our results with recently reported results in the literature. The questions that were used in our study can be accessed from Appendix A of the paper by Zhang and Pham [10].

We designed the survey using online survey development software (onlineanketler.com). We hosted our survey on this platform with the following address and made it available to participants for 1 month in January 2019: https://www.onlineanketler.com/s/99926a2. The results were recorded anonymously in the database of the hosting company. Participation in this survey research was voluntary. For the survey invitation, we sent e-mails to our network of professional contacts working in Turkish software organizations. We also shared our survey on social media (i.e. LinkedIn).

We selected 24 organizations from diverse industries to conduct this survey. Seventy survey forms were collected from 24 organizations, including the retail, energy, defense, aviation, financial technology (fintech), and banking sectors. We chose several organizations from various industries to allow a wide-ranging investigation, as discussed in Zhu et al.'s study [8]. This survey includes 32 environmental factors affecting reliability during the software development life cycle [10] as in the research by Zhu et al. [8]. The participants used a Likert scale (1–7) to score each factor. All the results were exported to an Excel file and processed using the techniques discussed in the next section.

3.3. Environmental factor analysis methods

Several methods were utilized to investigate the collected data, including relative weighted method, PCA, hypothesis testing (ANOVA), and correlation analysis. Before the investigation process started, all the data was normalized. Normalization was performed because some of the participants might give very high scores for many factors and we can get rid of this bias by normalizing the original ranking scores as discussed in Zhang and Pham's study [10].

Relative weighted method is used to rank the normalized data and determine the relative weights of factors based on only the participants' opinions. PCA is used to reduce the dimensions of a data set by providing key principal components. ANOVA is used to rank and determine the relative weights of factors depending on the information in the survey forms. Correlation analysis is applied to determine the strength of the relationship between variables, while the Pearson product-moment correlation coefficient (aka Pearson's r) is used to observe the relation between variables.

3.3.1. Relative weighted method

Relative weighted analysis is used to investigate the relative significance of each correlated component in the data. The main idea of relative weights analysis is that the correlated components are converted into new variants that are uncorrelated with one another but correlated with the respective original component (or predictor) variant, maximally [20]. The relative weighted method is implemented to unveil the ranking between the environmental factors in the present study. Let rij be the score of the ith factor in the jth survey. Firstly, we normalize rij for each survey by utilizing equation 1:

$$w_{ij} = \frac{r_{ij}}{\sum_{i=1}^{n} r_{ij}}. \tag{1}$$

The parameter N is the number of factors that are answered by the jth participant in the jth survey. Then equation 2 is used to acquire the final weight of the ith factor depending on average wij. l is the number of completed survey forms.

$$w_i = \frac{\sum_{j=1}^{l} w_{ij}}{l} \tag{2}$$

3.3.2. Principle component analysis

PCA is a mathematical method of explaining the information in a multivariate data set with fewer variables and minimal information loss. In other words, it is a transformation technique that reduces the size of the data set containing a number of correlated variables to smaller size components. PCA mainly reduces the dimensionality of data size. It is used to analyze the environmental factors.

3.3.3. Hypothesis testing

Hypothesis testing is implemented as a statistical hypothesis. It is a method of statistical interference that is testable depending on observation of a process modeled through a set of random variables. ANOVA is a statistical hypothesis testing method. It determines whether the population means of groups are equal or not in hypothesis testing and it explains the observations. In other words, it is utilized to simply analyze the differences among means of groups in a data set. We used the ANOVA with Tukey grouping to create the final grouping among the factors according to the means.

3.3.4. Correlation analysis

Correlation analysis aims to find out the correlation among the factors. The variables (factors) are correlated when the movement of one variable is accompanied by the movement of another variable. With the help of this approach, the dependent and independent variables can be determined.

4. Experimental results

During our survey, the participants stated that they had distinct positions such as software engineer, test engineer, architect, manager, and business analyst in their organizations as shown in Table 2. The category 'Others' covers the following positions: researchers, system administrators, product owners, and other positions in a software organization. According to our analysis, we noted that the number of experienced practitioners was larger than the number of nonexperienced ones who contributed to this survey. Based on this table, we see that most of the participants are software engineers and software architects. As the second characteristic, experience is represented in this table. Most of them have over 10 years of experience and therefore our results reflect the experience of the participants and only 17.1% of participants have less than 5 years of experience. The third characteristic is the company size and most of the participants work in organizations that have over 300 participants. The fourth and the last characteristics are about the reuse of software modules and the percentage of time spent in several phases of the software development life cycle.

Table 2. Characteristics of the participants in the survey.

Characteristic	Position	Mean score	Sample size
1. Current job positions	Software engineer	32.9%	23
	Software architect	24.3%	17
	Manager	12.9%	9
	Project manager	11.4%	8
	Others	8.6%	6
	Test engineer	5.7%	4
	Business analyst	4.3%	3
2. Experience	16+ years	31.4%	22
	15–11 years	28.6%	20
	6–10 years	22.9%	16
	0–5 years	17.1%	12
3. Company size	300–1000+	60.0%	42
	100–300	17.1%	12
	1–100	22.9%	16
4. Reuse of modules	300–1000+	54.8%	42
	100–300	50.9%	12
	1–100	50.0%	16
5. Time spent	Analysis	18.36%	88
	Design	16.5%	88
	Coding	43.2%	88
	Testing	18.4%	88

The outcome of the relative weighted method is presented in Table 3. Depending on the relative weights, the final weight for each factor can be obtained from Table 3. The environmental factor, which has a higher normalized weight value, impacts software reliability more than the factor having a lower normalized weight value. According to our analysis and Table 3, the top three important environmental factors are testing coverage, testing effort, and testing environment. All of these factors address the testing phase of software development.

Zhu et al. [8] reported that the top three factors are frequency of program specification change, testing effort, and testing environment. The only difference is the rank of frequency of program specification change. According to our analysis, it was at rank 7 among the 32 environmental factors.

A reason for this difference between our findings and other researchers' findings might be related to the software application domains covered by the participants, and also software development approaches followed within the organization. Zhu et al. [8] stated that software developers in their survey mainly focus on safety-critical applications. For safety-critical applications, defining the correct specifications is a critical component of the software development life cycle (SDLC), and sometimes this requires the use of formal specification languages. As such, those participants might have considered that the frequency of program specification change has a higher impact on software reliability. In contrast, developers from other domains in our survey might have regarded that the evolution of specifications can be tolerated with the help of best practices in agile software development approaches. Although SDLCs followed by the participants are not specified in Zhu et al.'s study [8], it is highly possible that traditional SDLCs (i.e. waterfall) might have been used because this type of SDLC is widely used for safety-critical system development. For this reason, those participants might have considered that the frequency of program specification change is one of the top environmental factors for software reliability.

The environmental factors that belong to the hardware category have lower normalized weights in our study and in Zhu et al.'s study [8]. The weakest factors in our study are shown as processors, storage devices, input/output devices, programming language, and telecommunication devices. In Zhu et al.'s study [8], we see similar factors, namely processors, telecommunication devices, system software, input/output devices, and storage devices. Based on Table 3, we clearly see that especially software testing and the other aspects of software development impact software reliability.

After this analysis, we wished to analyze whether there are correlated factors or not because the dimension of the factors can be reduced if there is a correlation among factors [8]. We utilized IBM SPSS to perform the PCA.

PCA can identify the critical principle components from a larger data set and therefore these critical factors can be checked by the software developers. Zhu et al. [8] selected the top 10 most important factors based on the relative weighted method for the PCA. We preferred the top 15 factors not to eliminate the other important factors for our PCA. Therefore, we selected the top 15 most important environmental factors (EFs) (i.e. f25, f22, f21, f1, f19, f18, f8, f15, f23, f5, f20, f24, f11, f27, and f6) depending on the relative weights listed in Table 3.

We selected the first five components based on the eigenvalues listed in Table 4. The eigenvalues of these five components are larger than 1.0 (values: 3.866, 1.859, 1.551, 1.348, and 1.158) and they cover 65.219% of the selected EFs. This means that about 65% of the variation can be explained by the first five components. Zhu et al. [8] determined the first three components and stated that 69% of the variation is explained by these three components. In their study, they referred to these new principles as overall, specification & knowledge, and program complexity & skill level. We had to select five components because only 48% of the variation can be explained by three components based on Table 4 and therefore we added two more components to reach the same level that was reported by Zhu et al. [8].

The correlated EFs with principle components for our analysis are listed in Table 5. Here we can identify the EFs regarding the new principles, namely overall, program workload & specification, reuse of software modules, requirement analysis & experience, and program complexity. This observation indicates that we

Table 3. Environmental factors ranking based on the relative weighted method.

Rank	EF	Description	Normalized weight	Type
1	f25	Testing coverage	0.038697968	Testing
2	f22	Testing effort	0.038199173	Testing
3	f21	Testing environment	0.037951869	Testing
4	f1	Program complexity	0.03645937	General
5	f19	Domain knowledge	0.035963725	Coding
6	f18	Program workload (stress)	0.035628719	Coding
7	f8	Frequency of program specification change	0.035111115	Analysis
8	f15	Average number of years in software development	0.034935078	Coding
9	f23	Testing resource allocation	0.034762455	Testing
10	f5	Level of programming technologies	0.034368275	General
11	f20	Human nature	0.034351975	Coding
12	f24	Testing methodologies	0.033557545	Testing
13	f11	Requirement analysis	0.033079452	Analysis
14	f27	Documentation	0.032980154	Testing
15	f6	Percentage of reused modules	0.031783889	General
16	f26	Testing tools	0.031746861	Testing
17	f32	System software	0.031466189	Hardware
18	f14	Development management	0.031336825	Analysis
19	f3	Difficulty of programming	0.031278877	General
20	f16	Number of years of software development more than 6	0.030977747	Coding
21	f13	Work standards	0.030668416	Analysis
22	f4	Amount of programming effort	0.030403282	General
23	f12	Relationship of detailed design and requirement	0.030376534	Analysis
24	f2	Program categories	0.029467065	General
25	f10	Design methodology	0.028837481	Analysis
26	f9	Volume of program design documents	0.028513439	Analysis
27	f17	Development team size	0.028218933	Coding
28	f31	Telecommunication devices	0.025018979	Hardware
29	f7	Programming language	0.024901915	General
30	f30	Input/output devices	0.024353444	Hardware
31	f29	Storage devices	0.023794677	Hardware
32	f28	Processors	0.023307416	Hardware

retain five principle components (PC1, PC2, PC3, PC4, and PC5). The loading coefficient shown in this table indicates that the corresponding factor has little impact if it is very small. For example, testing methodologies has the highest correlation with the first principle component and the requirement analysis has the highest correlation with the fourth principle component.

The reason for having two additional principles (i.e. reuse of software modules and requirement analysis) in our survey study might be related to the software application domains covered by the participants in

each survey study. While software reuse is inevitable in business application domains, the reuse of software modules in safety-critical applications might be limited due to the complexity of requirements and projects. As explained above, the previous survey study stated that developers mainly focus on safety-critical applications, and therefore those participants might not have considered the reuse of software modules as a critical issue for software reliability. In our survey, developers might have focused on business applications, and, as such, the reuse of software modules might be more visible for them. In our study, the new principle was requirement analysis because its importance is nowadays highly appreciated by software developers. For the previous survey, the specification & knowledge was considered as a principle. From the safety-critical application developer perspective, program specification might have been viewed as more important than the requirement analysis.

Table 4. Eigenvalue of the correlation matrix.

	Initial eigenvalues			Rotation sums of squared loadings		
PCA	Total	% of variance	Cumulative %	Total	% of variance	Cumulative %
1	3.866	25.777	25.777	3.866	25.777	25.777
2	1.859	12.393	38.169	1.859	12.393	38.169
3	1.551	10.339	48,508	1.551	10.339	48.508
4	1.348	8.989	57.497	1.348	8.989	57.497
5	1.158	7.723	65.219	1.158	7.723	65.219
6	0.903	6.022	71.241			
7	0.716	4.772	76.013			
8	0.646	4.310	80.323			
9	0.628	4.187	84.510			
10	0.575	3.833	88.342			
11	0.471	3.139	91.481			
12	0.436	2.909	94.390			
13	0.379	2.525	96.915			
14	0.260	1.733	98.648			
15	0.203	1.352	100.000			

After the PCA, we aimed to analyze whether these environmental factors have the same effect on software reliability or not by using ANOVA. The environmental factors were investigated to compare their significance levels for software reliability. The results of this method are shown in Table 6. Based on the ANOVA (Tukey) method, 16 groups were identified. In Zhu et al.'s study [8], 9 groups were identified. In Table 6, we show the environmental factors from the greatest to the least significance on software reliability and, based on this table, the most significant factor is the testing coverage and then comes the testing effort. Testing environment is the third most important factor. In Zhu et al.'s study [8], the most significant factor was the testing effort factor and then came the frequency of program specification change. The third most significant factor was the testing environment. Except for the frequency of program specification change factor, the top three significant factors are the same in these two studies. According to our analysis, factors between average number of years in software development and testing tools belong to the same significant level.

The basis for this different finding in the top three significant factors might be related to the software application domains addressed by the participants in the two studies. In Zhu et al.'s study [8], it was reported

Table 5. New principles for each principle component.

Component	New principle	Environmental factor	Loading coefficient
PC1	Overall	f24-Testing methodologies	0.722
		f22-Testing effort	0.665
		f25-Testing coverage	0.665
		f19-Domain knowledge	0.615
		f5-Level of programming technologies	0.604
		f27-Documentation	0.584
		f21-Testing environment	0.561
		f20-Human nature	0.493
		f23-Testing resource allocation	0.475
PC2	Program workload & specification	f18-Program workload (stress)	0.759
		f8-Frequency of program specification change	0.662
PC3	Reuse of modules	f6-Percentage of reused modules	0.470
PC4	Requirement analysis & experience	f11-Requirement analysis	0.460
		f15-Average number of years in software development	0.456
PC5	Program complexity	f1-Program complexity	0.634

that participants mainly focus on safety-critical applications. Therefore, the frequency of program specification change might have been considered one of the top significant factors for software reliability. However, participants who focused on business applications in our survey study might have regarded that testing coverage is more crucial than program specification because formal program specification is more common in safety-critical application domains.

After the ANOVA, we aimed to analyze the relation between variables and determine the strength of this relationship. Therefore, we performed correlation analysis and calculated the Pearson product-moment correlation coefficient (aka Pearson's r). This coefficient is between –1 and 1. While 1 indicates a positive correlation, 0 means no relationship exists and –1 shows that there is a total negative correlation. Correlations of EFs are provided in the Supplementary Material due to the page limit for the article. Correlation is significant at 0.01 level (2-tailed).

After we completed the environmental factor analysis, we started to work on the development life cycle phase analysis as in the study by Zhu et al. [8]. Zhang and Pham [10] categorized the environmental factors into the following five groups: general, analysis & design, coding, testing, and hardware systems. In the present study, we aimed to analyze whether the first four groups of factors have the same significant levels on software reliability or not. We applied ANOVA to investigate the impact level of development phases and the results are shown in Table 7. In this table, the mean score is from 4.9385 to 5.6263. To group development phases based on the mean score, the Tukey grouping method was applied. Three final groups were identified as in the study by Zhu et al. [8].

Table 6. Final grouping based on the ANOVA (Tukey) method.

Factor	N	Mean	Final group	Grouping								
f25-Testing coverage	70	6.114	1	A								
f22-Testing effort	70	6.057	1	A								
f21-Testing environment	70	5.986	2	A	B							
f19-Domain knowledge	70	5.729	3	A	B	C						
f 1-Program complexity	70	5.714	3	A	B	C						
f18-Program workload (stress)	69	5.638	4	A	B	C	D					
f 8-Frequency of program specification change	68	5.544	5	A	B	C	D	E				
f15-Average number of years in software development	70	5.5	6	A	B	C	D	E	F			
f 5-Level of programming technologies	70	5.471	6	A	B	C	D	E	F			
f23-Testing resource allocation	70	5.471	6	A	B	C	D	E	F			
f20-Human nature	70	5.429	6	A	B	C	D	E	F			
f24-Testing methodologies	69	5.348	6	A	B	C	D	E	F			
f11-Requirement analysis	67	5.313	6	A	B	C	D	E	F			
f27-Documentation	70	5.271	6	A	B	C	D	E	F			
f26-Testing tools	68	5.118	6	A	B	C	D	E	F			
f32-System software	69	5.029	7		B	C	D	E	F	G		
f 6-Percentage of reused modules	70	5	8		B	C	D	E	F	G	H	
f14-Development management	69	5	8		B	C	D	E	F	G	H	
f 3-Difficulty of programming	70	4.914	9			C	D	E	F	G	H	
f13-Work standards	69	4.913	9			C	D	E	F	G	H	
f16-Number of years of software development more than 6	70	4.9	9			C	D	E	F	G	H	
f12-Relationship of detailed design and requirement	69	4.87	9			C	D	E	F	G	H	
f 4-Amount of programming effort	70	4.786	10			C	D	E	F	G	H	I
f 2-Program categories	69	4.681	11				D	E	F	G	H	I
f10-Design methodology	67	4.597	12					E	F	G	H	I
f 9-Volume of program design documents	67	4.537	12					E	F	G	H	I
f17-Development team size	70	4.5	13						F	G	H	I
f31-Telecommunication devices	65	4.077	14							G	H	I
f 7-Programming language	69	3.986	15								H	I
f30-Input/output devices	66	3.985	15								H	I
f29-Storage devices	70	3.843	16									I
f28-Processors	70	3.814	16									I

The testing phase is in group 1, the coding phase is in group 2, and the analysis & design and general phases are in group 3. This result is slightly different than the grouping reported by Zhue et al. [8], because in their study only the general phase is in group 3 and in group 2 there are the analysis & design and coding phases.

This shows that the analysis & design phase had one level higher importance in their survey study. This small difference might be related to the application domains, which are focused on by participants, and safety-critical software developers might have considered analysis & design to be of greater importance compared to the other developers in our survey. The final grouping table shows that the testing phase has the highest mean value, which means that the testing phase is the most significant phase in software reliability.

As shown in Table 3, only four environmental factors in the top 10 most important environmental factors belong to the testing phase, which indicates that those four environmental factors are in the top 10 list. This observation is consistent with the study by Zhu et al. [8], because they also reported that the testing phase covers 40% of the factors in the top 10 list. Based on this analysis, we can state that software testing is still the most crucial phase of the software development process.

We were also interested in checking whether the opinion of people depends on the level of experience and the organizational size (i.e. number of people in the organization). A similar analysis was performed by Zhang et al. [11] and they reported that participants who have different levels of experience had different opinions while ranking the factors. As such, we divided participants into four experience categories (0–5 years, 6–10 years, 11–15 years, and 16+ years) based on our participants and checked their opinions regarding the importance of development phases. Table 8 presents the analysis between the experience categories and the development phase.

As shown in Table 8, these participants have a similar perspective on the required time that must be spent for each development phase. However, experienced participants place slightly more emphasis on the coding phase. For the organizational size effect analysis, we divided the organizations into three categories based on the number of people (1–100 people, 100–300 people, and 300+ people) in the organization. Table 9 shows the analysis between the company size and the required time that must be allocated per development phase. As shown in Table 9, participants who work in large companies prefer spending more time on the testing phase.

Table 7. Final grouping based on the ANOVA (Tukey) method for the development phase.

Factor	N	Mean	Final group	Grouping		
Testing	487	5.6263	1	A		
Coding	419	5.2816	2		B	
Analysis & design	476	4.9685	3			C
General	488	4.9385	3			C

Table 8. Analysis of the personnel experience for ranking the development phases.

Experience	Analysis	Design	Coding	Testing
16+ years	16.6	20.2	43.6	19.5
11–15 years	16.4	20.5	43.3	19.8
6–10 years	16.8	21.0	42.5	19.8
0–5 years	16.8	21.0	42.5	19.8

Table 9. Analysis of the organizational size for ranking the development phases.

Size (# of people)	Analysis	Design	Coding	Testing
300+	18.4	17.3	43.5	20.7
100–300	14.1	18.2	50.0	17.7
1–100	21.4	18.9	55.9	18.1

5. Discussion

In this section, we discuss whether we observed any change in the significance rankings of environmental factors or not and compare our findings with those of other studies reported in the literature. This kind of comparison is quite useful for organizations and developers because up-to-date factors help to improve software reliability and increase efficiency. It is also possible to integrate these up-to-date factors into software reliability models for better software reliability analysis [8]. In Table 10 we show that we included a sufficient number of participants, used similar environmental factors, and applied similar techniques.

Table 10. Summary of our study.

Study	Our study
Data	70 surveys (24 organizations)
Factors	32
Method	Relative weighted method and ANOVA
Measurement scale	Likert (1–7)
Factor ranking	Normalized data with relative weighted method
Factor classification	Yes (Tukey method)
Correlation analysis	Yes (dependency between factors)
Dimensionality reduction	PCA
Dev. phases grouping	Yes

The populations in our study and the study by Zhu et al. [3] are from two different countries and the survey times are also different. Therefore, the results of these two studies with respect to rankings are not the same. We observed that most of the environmental factors in the top 10 list reported by Zhu et al. [8] are still in the 10 top list except for three, namely percentage of reused modules (f6), relationship of detailed design to requirement (f12), and testing methodologies (f24). The new factors that are now in the top 10 list are the program workload (f18), testing resource allocation (f23), and the level of programming technologies (f5). Testing coverage (f25) is the most important factor based on our survey study and this factor was at rank 4 in Zhu et al.'s study [8].

While Zhu et al. [8] selected the top 10 important factors for PCA and Zhang and Pham [11] preferred the top 11 factors, we selected the top 15 factors for factor analysis. Zhu et al. [8] presented three new principles, Zhang and Pham [11] showed four principles, and we reached five principles. The principle components in our study (overall, program workload & specification, reuse of software modules, requirement analysis & experience, and program complexity) can be considered as the consolidation of the two studies because Zhu et al. [8] reported the overall, specification & knowledge, and program complexity & skill level principles and Zhang and Pham [11] determined the overall, testing efficiency, requirements & specifications, and program & skill level principles.

Based on the Tukey method, our result and the result reported by Zhu et al. [8] show that there are three final groups. As in Zhu et al.'s study, the testing phase and general phase are in two different groups. Testing is still the most important phase. Analysis & design in our study is in group number 3, but it was in group 2 in Zhu et al.'s study [8]. This result shows that the participants in our study put more emphasis on the coding phase compared to the analysis & design phase. This is an important observation for global software engineering projects because participants in the study by Zhu et al. [8] put higher emphasis on analysis & design than the emphasis in our study. While the two populations appreciate the importance of the testing phase and agree that it is the most important phase of the software development process, there is a change in the order of the other phases.

The time allocation for each phase in Zhu et al.'s study [8] was reported as follows: 22% analysis phase, 20% design phase, 34% coding phase, and 24% testing phase. As shown in Table 8, the percentage of the time allocation for the coding phase is higher (42%-43%) than that for the coding phase (34%) of Zhu et al.'s study [8]. The percentage of the time spent on design is quite similar. The times spent on the analysis and testing phases are not as high as in Zhu et al.'s study [8] (approx. 5% less).

We discuss the following four potential threats to validity [21–23]: external validity, conclusion validity, construct validity, and internal validity. Conclusion validity threats are those affecting the ability to conclude appropriately. Construct validity threats are those impacting the fused data validation. External validity threats are those limiting the study's generalization. The relationship between the outcome and treatment must be causal, but if it is related to a factor that cannot be controlled by the researchers, it is said that the internal validity is low. Regarding the external validity, we worked with 70 volunteer software professionals from 24 organizations in Turkey and they represent different software organizations focusing on different application domains. If we had worked with other groups of software experts, our results would have been different than those reported in the present study. Therefore, observations might be different on a data set including different sets of volunteer participants. However, since we do not work only with professionals from a single organization, our results should be generalized outside of this context. Regarding the conclusion validity, we applied several state-of-the-art statistical analysis techniques such as PCA and ANOVA to evaluate the collected data from several perspectives and reported our observations based on the results of these techniques. Since we also followed the same protocol and analysis techniques discussed in Zhu et al.'s study [8], our results can be compared with those reported by Zhu et al. [8]. Regarding the construct validity, there might be threats related to the misunderstandings between participants and researchers in the present study. However, we explained the goal of our study at the beginning of the survey form and responded to participants who had questions via e-mail. Regarding the internal validity, we must address criterion validity as this research included questionnaires and, in the context of questionnaires, criterion validity is used to check if the items in the questionnaire actually measure real-world events. We applied the same set of items used by Zhu et al. and Zhang et al. [8, 10, 11], and so we do not see any threat regarding internal validity. All the items in the questionnaire measure the real-world states.

6. Conclusion and future work

We conducted a survey study to investigate the impact level of environmental factors among software professionals in Turkey and compared the results with those of previously published studies. Most of the environmental factors shown in the top 10 list in Zhu et al.'s study [8] are still in the top 10. However, there are some changes for some factors. Program workload (f18), testing resource allocation (f23), and level of programming technolo-

gies (f5) are new factors that are added to the top 10 list. This shows that we have more testing factors in the top 10 list and programming technologies impact software reliability. Five principle components, PC1, PC2, PC3, PC4, and PC5, as shown in Table 5, can represent 65.219% of variation in all the factors. These five components are different from the three principle components reported in Zhu et al.'s study [8]. We demonstrated that software testing is still the most important factor for software reliability. The coding phase ranked second in our study, but it was listed after the analysis & design phase in Zhu et al.'s study [8]. The time allocation regarding each phase in the software development process is different compared to that in the study by Zhu et al. [8]. To the best of our knowledge, ours is the first study to compare the impact of environmental factors of software reliability in a country different from that in the initial studies. It is highly interesting to examine the different interpretations of software experts from different cultures on software reliability.

References

[1] Friedman MA, Voas JM. Software Assessment: Reliability, Safety, Testability. New York, NY, USA: John Wiley & Sons, 1995.

[2] Febrero F, Calero C, Moraga M. A systematic mapping study of software reliability modeling. Information and Software Technology 2014; 56 (8): 839-849.

[3] Zhu M, Pham H. A two-phase software reliability modeling involving with software fault dependency and imperfect fault removal. Computer Languages, Systems & Structure 2018; 53: 27-42.

[4] Utkin V, Coolen FPA. A robust weighted SVR-based software reliability growth model. Reliability Engineering & System Safety 2018; 176: 93-101.

[5] Yazdanbakhsh O, Dick S, Reay I, Mace E. On deterministic chaos in software reliability growth models. Applied Soft Computing 2016; 49: 1256-1269.

[6] Wang H, Fei H, Yu Q, Zhao W, Yan J et al. A motifs-based maximum entropy Markov model for realtime reliability prediction in system of systems. Journal of Systems and Software 2019; 151: 180-193.

[7] Organizaci O. ISO-IEC 25010: 2011 Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation (SQuaRE)-System and Software Quality Models. 2011.

[8] Zhu M, Zhang X, Pham H. A comparison analysis of environmental factors affecting software reliability. Journal of Systems and Software 2015; 109: 150-160.

[9] Zhu M, Pham H. Environmental factors analysis and comparison affecting software reliability in development of multi-release software. Journal of Systems and Software 2017; 132: 72-84.

[10] Zhang X, Pham H. An analysis of factors affecting software reliability. Journal of Systems and Software 2000; 50 (1): 43-56.

[11] Zhang X, Shin MY, Pham H. Exploratory analysis of environmental factors for enhancing the software reliability assessment. Journal of Systems and Software 2001; 57 (1): 73-78.

[12] Mishra A, Yazici A, Cetin S. Software evolution in Turkey. Tehnicki Vjesnik 2016; 23 (3): 929-935.

[13] Loganathan A, Muthuraj RJ. A new methodology for data reduction in software reliability studies. Communications in Statistics: Case Studies, Data Analysis and Applications 2016; 2 (3-4): 101-105.

[14] Pfleeger SL, Kitchenham BA. Principles of survey research: part 1: turning lemons into lemonade. ACM SIGSOFT Software Engineering Notes 2001; 26 (6): 16-18.

[15] Kitchenham BA, Pfleeger SL. Principles of survey research part 2: designing a survey. ACM SIGSOFT Software Engineering Notes 2002; 27 (1): 18-20.

[16] Kitchenham BA, Pfleeger SL. Principles of survey research part 6: data analysis. ACM SIGSOFT Software Engineering Notes 2003; 28 (2): 24-27.

[17] Kelley K, Clark B, Brown V, Sitzia J. Good practice in the conduct and reporting of survey research. International Journal for Quality in Health Care 2003; 15 (3): 261-266.

[18] Aday LA, Cornelius LJ. Designing and conducting health surveys: a comprehensive guide.San Francisco, CA, USA: Jossey-Bass, 2006.

[19] Bowling A. Research Methods in Health: Investigating Health and Health Services. Maidenhead, UK: McGraw-Hill Education; Open University Press, 2014.

[20] Johnson JW. A heuristic method for estimating the relative weight of predictor variables in multiple regression. Multivariate Behavioral Research 2000; 35 (1): 1-19.

[21] Wohlin C, Runeson P, Host M, Ohlsson MC, Regnell B, Wesslen A. Experimentation in Software Engineering: an Introduction. Norwell, MA, USA: Kluwer, 2000.

[22] Easterbrook S, Singer J, Storey MA, Damian D. Selecting empirical methods for software engineering research. In: Shull F, Singer J, Sjøberg DIK (editors). Guide to Advanced Empirical Software Engineering. London, UK: Springer, 2008, pp. 285-311.

[23] Zhou X, Jin Y, Zhang H, Li S, Huang X. A map of threats to validity of systematic literature reviews in software engineering. In: 2016 23rd Asia-Pacific Software Engineering Conference (APSEC); New York, NY, USA; 2016. pp. 153-160.