

Received 25 October 2022, accepted 13 November 2022, date of publication 17 November 2022, date of current version 23 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3222845

## RESEARCH ARTICLE

# A Quantitative Assessment of the Impact of Homogeneity in Personality Traits on Software Quality and Team Productivity

NOSHEEN QAMAR<sup>1</sup> AND ALI AFZAL MALIK<sup>2</sup>, (Senior Member, IEEE)

<sup>1</sup>Department of Software Engineering, University of Management and Technology, Lahore 54700, Pakistan

<sup>2</sup>Department of Computer Science, National University of Computer and Emerging Sciences, Lahore 54700, Pakistan

Corresponding author: Nosheen Qamar (nosheen.qamar@umt.edu.pk)

**ABSTRACT** Software industry is increasingly focusing on the improvement of software quality and team performance to survive in a competitive and rapidly growing environment. Previous studies reveal the importance of social and human factors in software engineering. Despite the importance of these personality factors, only a handful of empirical studies have quantitatively evaluated the impact of personality traits on software quality and effectiveness of software engineering teams. This study quantifies the abstract notion of trait-wise team homogeneity based on the Five-Factor Model (FFM). Additionally, this study also aims to measure its impact on team productivity and software quality. Trait-wise team homogeneity measures the Team Homogeneity Index (THI) for each of the five personality traits of FFM. Therefore, five new metrics i.e. Openness THI (O-THI), Conscientiousness THI (C-THI), Extraversion THI (E-THI), Agreeableness THI (A-THI), and Neuroticism THI (N-THI) are proposed. The utility of these five metrics is evaluated by conducting experiments in academic and industrial environments for three different phases of the Software Development Life Cycle (SDLC) i.e. analysis and design, implementation, and testing. During the analysis and design phase, it was observed that teams (composed of either students or professionals) with greater values of C-THI and A-THI produced better quality design models and teams with greater values of C-THI were more productive. In the implementation phase, teams with higher values of O-THI, C-THI, and E-THI were noticeably more productive and produced better quality code. For the testing phase, teams with greater values of C-THI and A-THI produced better quality test cases and were more productive. Results obtained so far indicate that the newly proposed five metrics – O-THI, C-THI, E-THI, A-THI, and N-THI – appear to have the potential to aid managers and academics alike in building productive teams which can produce better quality software.


**INDEX TERMS** Personality profiling, personality traits, software quality, team homogeneity, team personality, team productivity, trait-wise team homogeneity.

## I. INTRODUCTION

Software development organizations are increasingly focusing on innovative and creative ideas to remain alive in this age of cut-throat competition and rapid growth in the software industry. Due to this competitive environment and pace of growth, software development has become more challenging, complex, and demands diverse competencies from software

developers. Qualification and technical skills alone are not enough to guarantee project success [1]. Soft skills also play an important role in influencing software quality and team productivity which are essential ingredients for a successful software development project.

Software quality (i.e. degree to which software meets client's expectations/requirements [2]) and team performance are heavily influenced by human aspects e.g. cooperation, communication, negotiation, decision making, etc. [3]. Furthermore, software quality and team productivity (i.e.

The associate editor coordinating the review of this manuscript and approving it for publication was Francisco J. Garcia-Penalvo .

rate of software produced per unit of cost to produce it [2]) can be improved by composing teams with the right people [4], [11].

The value of human factors in software development teams cannot be overlooked. According to Pressman and Maxim [2], software is developed by people, used by people, and needs interaction between people. Similarly, DeMarco and Lister clearly mention that, “Most software development projects fail because of failures with the team running them” [4]. Even COCOMO, which is the most widely used effort estimation model, also takes personnel attributes (i.e. capabilities and experience) into account [5]. However, despite the significance of human aspects, most of the past research has focused on technical factors in software development [6].

Despite their importance, relatively few studies [e.g. 7-12] have investigated the significance of human factors in software engineering. The focus of these studies was mainly on just exploring the impact of team members’ personalities on different aspects of software engineering e.g. decision making, work habits, team satisfaction, work climate, software quality, team productivity, etc. This indicates that a relation exists between personality and team performance. That is why, a deeper understanding of personality and analysis of its impact on team performance and product quality is vital in building an effective team [13].

Personality is one of the most important human factors and, therefore, greatly influences the team’s performance [7], [8], [9], [10], [11], [12], [13]. According to Pervin, personality can be defined as a person’s enduring, idiosyncratic, and distinctive attributes that make a person different from others [14]. Different models have been proposed to assess the personality of an individual. The three most widely used personality assessment models are Myers-Briggs Type Indicator (MBTI) [15], Five-Factor Model (FFM) [16], and Keirsey Temperament Sorter (KTS) [17]. It was reported in 2015 that MBTI (which is the oldest personality model) has been used by 35% of past studies while FFM was used by 11% and KTS by 7% of past studies [18]. MBTI and KTS are based on personality type theory whereas FFM is based on personality trait theory.

Previous research [2], [7], [12], [13] indicate that an effective team with the right people can positively influence team productivity and software quality. Furthermore, all of the team members are not equally effective as some individuals outperform others and are more efficient [14]. Thus, it is important for software development managers to take human aspects into account while building their teams. Assembling a group based on the attributes of its members is called team composition [19]. Attributes of group members incorporate their culture, expertise, demographics, and other aspects concerning their personalities [23]. Past researchers [12], [20], [21], [22], [23] have also discovered the effects of heterogeneous and homogeneous personalities on team performance. Some researchers suggest that homogeneous teams perform better [12], [20], [21] than teams with diverse personalities. Others contend that heterogeneous

teams play an important role in improving team performance [22], [23].

Only a limited number of studies have focused on the impact of homogeneity/heterogeneity on team satisfaction, team performance, and software quality [12], [20], [21], [23]. Furthermore, only a couple of studies [11], [20] have quantitatively evaluated the team personality using a measure of central tendency (i.e. mean). Our previous work [12], was the first attempt to quantitatively evaluate the impact of team homogeneity on software quality and team productivity using a new metric called the Team Homogeneity Index (THI) which was based on spread rather than central tendency. Spread is considered a better representation of a dataset as spread focuses on variation or dispersion across the dataset. The mean just takes the central point into account to represent the dataset. For example, the mean of two datasets (1, 25) and (12, 14) is the same even though the range or dispersion in these datasets is different.

In this research, we have quantified (using the variance as a measure of spread) the abstract notion of trait-wise team homogeneity with the help of five new metrics, namely Openness THI (O-THI), Conscientiousness THI (C-THI), Agreeableness THI (A-THI), Extraversion THI (E-THI), and Neuroticism THI (N-THI). Additionally, we have evaluated the utility of these five newly proposed metrics by analyzing their impact on quality and team productivity during three different phases of the software development life cycle (SDLC) i.e. analysis and design, implementation, and testing. Furthermore, we have done a comparative analysis of the impacts of these five metrics to determine which trait correlates more with software quality and team productivity.

The remaining paper is organized as follows. The next section provides an overview of related work. Section 3 briefly describes the Five-Factor Model (FFM). The process of quantification of trait-wise team homogeneity is presented in Section 4. Section 5 provides the details of our experimentation and Section 6 summarizes the results along with a discussion about our main findings. Section 7 describes some potential threats to the validity of our results and Section 8 states the important implications of our research. Finally, we highlight our main conclusions and present some future research directions in Section 9.

## II. RELATED WORK

The effect of personality in software engineering has been evaluated by several researchers in academic and industrial environments [18], [24]. A systematic mapping study of the impact of personality in software engineering was conducted by Cruz et al. in 2015 [18]. The results of this study revealed that, after 2002, 83% of research articles which evaluated the impact of personality in software engineering were based on empirical findings [18]. The main emphasis of previous research was on analyzing the impact of personality on software quality, project success, team effectiveness, job satisfaction, team performance, etc. [1], [6], [18], [24], [62].

Pieterse et al. [23] used KTS to examine the impact of personality on the performance of the team. Their results

indicated that heterogeneous teams performed better than homogeneous teams, particularly during the earlier phases of team growth. Gulati et al. [25] also used KTS to conduct a study involving 66 students to assess the effect of personality temperaments on programmers' performance. They found a strong relationship between programmers' performance and Guardian temperament.

Sfetsos et al. [26] assessed the impact of personality composition on the performance of programmers involved in pair programming. They also used KTS as a personality profiling tool. It was discovered that pairs with diverse personality temperaments showed better pair collaboration, communication, and pair performance as compared to pairs with identical personality temperaments.

Karn et al. [27] evaluated the impact of MBTI personality types on team effectiveness in XP projects. Their results indicated that personality profiling of software engineering teams is very important. They also found that personalities have a significant impact on the performance of teams. Teams with homogeneous personalities were found to be more competitive. Choi et al. [28] used MBTI in their research. They conducted an empirical study by involving 68 undergraduate students and 68 master's degree graduate students to find out the impact of team homogeneity and heterogeneity on the productivity of teams in pair programming. Their results showed that the pairs with heterogeneous personalities were more effective than the pairs with like personalities.

Gilal et al. [29] conducted a study to evaluate the impact of MBTI personality types on the composition of software teams. They used three classification methods i.e. rough sets theory, decision tree, and logistic regression for the composition of software teams and suggested a model to predict team performance. The predictors used in this model include gender, personality type, and team role. The team composition method's effectiveness was assessed based on higher precision and reduced complexity. Their results showed that the Johnson Algorithm (JA) of rough set theory is the most effective technique for the team composition. Poonam and Yasser [30] analyzed the impact of MBTI personality types in two different situations involving pair programming. In the first situation, pairs performed tasks at the same location whereas in the second situation pairs worked from different places. Their results showed that personality has a greater influence on the performance of the pair working remotely as compared to pair working at the same location.

Kamangar et al. [31] conducted a study (using MBTI) to find the relationship between the personalities of software testers and the efficiency of testing methods. The outcomes of this study revealed that testers with introvert personality types were more efficient in white box testing, on the other hand, extroverts were more productive in black-box testing. Barroso et al. [32] carried out a study to examine the effect of students' personality types on software complexity measured with the help of the CK metrics suite [33]. MBTI was used for personality profiling of the students. Results of this study revealed that personality has a noteworthy impact on Depth of Inheritance Tree (DIT). No notable influence was observed

for Weighted Methods per Class (WMC) and Response For a Class (RFC) metrics.

Kichuk and Wiesner [20] conducted a study to assess the relationships between FFM personality factors and team performance for product design teams. Teams with higher agreeableness, higher extraversion, higher levels of cognitive ability, and lower neuroticism were found to be more successful. Peeters et al. [21] also conducted a study using FFM to examine the impact of team composition on team performance. Their results indicated that homogeneous teams were better in solving structured tasks whereas heterogeneous teams were more useful in solving unstructured tasks.

Walle and Hannay [34] evaluated the impact of personality traits on the collaboration and efficiency of a pair. Their results indicated that personality might affect pair collaboration and that the impact of personality on pair collaboration was more pronounced as compared to the impact on pair performance. Hannay et al. [35] further evaluated the effect of FFM personality traits on the programming skills of pair programmers. Their results indicated that to determine the connection between personality factors and pair performance more research should be done. Chao and Atli [36] assessed the effect of four different personality traits (i.e. Open-minded, Logical, Responsible, and Attentive) on the success of pair programming. Their results showed no correlation between code quality and the selected four personality traits.

Feldt et al. [37] conducted a study to analyze the relationship between the viewpoints and behavior of 47 professional software developers. They used FFM to assess the personality of the practitioners. Their results showed that software engineers having similar personality attributes have the same viewpoints. Kanij et al. [38] used FFM to assess the effect of personality attributes on software testers' performance. It was found that testers' performance had a positive correlation with the conscientiousness and extraversion personality attributes.

Salleh et al. [39] evaluated the impact of FFM personality traits (i.e. conscientiousness, openness to experience, and neuroticism) on pair programmers' performance by conducting four experiments. It was reported that neuroticism and conscientiousness do not have a significant impact on the pair programmers' performance. However, openness to experience considerably influenced the pair performance.

Acuña et al. [11] conducted a replicated study involving a large number of subjects to evaluate the impact of team climate and personality on job satisfaction and software quality. Their results revealed that a positive correlation exists between team climate factors and job satisfaction. Furthermore, a positive relationship was observed between extraversion and agreeableness personality traits and software quality.

Lykourantzou et al. [40] investigated the effect of a balanced/imbalanced team on the team's performance. They also used FFM for personality profiling. Imbalanced teams were composed of individuals with conflicting personalities whereas balanced teams were composed of individuals with harmonious personalities. Their results revealed that balanced

teams' performance was much better with higher cooperation and lower conflicts.

Yilmaz et al. [13] also carried out a survey of professionals. They used FFM as a tool for personality assessment. Their results indicated that a team with agreeableness, emotional stability, higher extraversion, and conscientiousness performed better than other teams.

Shameem et al. [41] recommended a framework by taking team climate and team members' FFM personality traits into consideration. They reported that with the help of their framework performance of teams can be improved. Furthermore, it was found that teams with individuals having extraversion and conscientiousness personality traits performed better.

Barroso et al. [42] assessed the effect of students' personality traits on software quality. A positive correlation between the cyclomatic complexity of software and three FFM personality traits (i.e. conscientiousness, openness, and neuroticism) was observed. Furthermore, agreeableness was found to be more significant in influencing the Coupling Between Objects (CBO) whereas neuroticism and extraversion were found to have a significant correlation with Depth of Inheritance Tree (DIT) and Coupling Between Objects (CBO) metrics of the CK metrics suite.

Amin et al. [43] analyzed the impact of the big five personality traits and knowledge collection behavior on a programmer's creativity intention. Their results revealed that programmer's personality traits and knowledge collection behavior play a key role in shaping their intention to be creative. Yan et al. [44] presented a novel analytic approach to extract and understand these behavioral patterns and their impact on predicting adaptive and maladaptive personality traits. Their machine learning analysis shows that both traits are associated with passively sensed behavior. Kosan et al. [45] created a personality dataset from twitter and unstructured data were transformed into meaningful and processable data, LSTM-based prediction models were created with the structural analysis, and evaluations were made on both dataset and PAN-2015-EN.

The overwhelming majority of previous studies were qualitative in nature. Only a few [11], [20], [62] evaluated the impact of personality in software engineering quantitatively and that too using the mean (a measure of central tendency). Our previous work [12], [62] filled this research gap by quantitatively evaluating (using the variance as a measure of spread) the impact of team homogeneity on team productivity and software quality. In our first study [12], we proposed a metric called Team Homogeneity Index (THI) and evaluated its impact on software quality and team productivity for two phases (implementation and testing) of the SDLC. In a follow-up study [62], we replicated our experiment on three different phases of the SDLC (i.e. analysis and design, implementation, and testing) and determined the weights for all five personality traits using input from the industry. We proposed an improved version of Team Homogeneity Index (THI) called Weighted Team Homogeneity Index (WTHI). Furthermore, we also conducted a comparative analysis of THI and WTHI to determine whether weights assigned to personality

traits make any difference. This research builds on our previous work by proposing five additional metrics each of which is based on one of the traits of FFM (i.e. Openness THI (O-THI), Conscientiousness THI (C-THI), Extraversion THI (E-THI), Agreeableness THI (A-THI), and Neuroticism THI (N-THI)). The calculation process of trait-wise team homogeneity indices is different from that of Team Homogeneity Index (THI). In case of THI, we measure the overall homogeneity within all five traits. In the case of trait-wise team homogeneity, on the other hand, we calculate the homogeneity between team members for each trait. Table 1 provides a summary of the past work related to personality and software teams.

### III. FIVE FACTOR MODEL - BACKGROUND

The Five-Factor Model (FFM), which is also called "Big Five" is a classification of personality traits. It consists of five main personality traits i.e. openness, conscientiousness, extraversion, agreeableness, and neuroticism (as shown in Figure 1). These traits provide a structure of human personality by dividing it into five different dimensions [16]. These five traits belong to the trait theory. American Psychiatric Association defines a "trait" as the lasting patterns of perceiving, associating to, and thinking about the surroundings and oneself. Each of the five traits of FFM is defined as follows [16]:

Openness relates to cultural, creative, and intellectual interest. People with a high score in openness are likely to be open-minded, curious, and imaginative, whereas people with low openness score typically show a preference for routine, favoring conservative values, and aesthetic sensibilities.

Conscientiousness focuses on an individual's success orientation. Individuals having a high score of conscientiousness are likely to be organized, perseverant, hardworking, scrupulous, responsible, and reliable. In contrast, people with low conscientiousness tend to be impulsive, disordered, and irresponsible.

Extraversion refers to the extent of activeness, assertiveness, sociability, talkativeness, and gregariousness. An individual with a high score of extraversion feels comfort in social affiliation, active, friendly, outgoing, and assertive whereas an individual with a low extraversion score tends to be quiet, reserve, passive, and likes to stay alone.

Agreeableness relates to attributes such as trust, cooperativeness, warmth, and kindness. Someone who has a high agreeableness score appears as good-natured, soft-hearted, and cooperative. On the other hand, individuals with low agreeableness scores tend to be hostile, suspicious, selfish, ruthless, and skeptical.

Neuroticism focuses on the state of emotional stability. People with high neuroticism have a habit of being nervous, moody, insecure, and anxious whereas an individual with low neuroticism is inclined to appear confident, secure, comfortable, and calm.

FFM is operationalized using different instruments e.g. NEO Five-Factor Inventory (NEO-FFI) [46], Revised NEO Personality Inventory (NEO-PI-R) [47], International

**TABLE 1. Summary of empirical studies related to personality and software teams.**

Sr#	Author(s) Ref.]	Personality Model	Study Type	Subjects (Size)	Outcomes/Results	
1	Pieterse & Kourie [23]	KTS	Exp	Stu (12 Teams)	Diverse teams performed better than homogeneous teams	
2	Gulati et al. [25]		Exp	Stu (66)	A strong relationship has been found between programmers' performance and Guardian temperament	
3	Sfetsos et al. [26]		Exp	Stu (84)	Pairs with diverse personalities performed better because they communicated and collaborate more	
4	Karn et al. [27]	MBTI	Exp	Stu (5 Teams)	Homogeneous teams proved to be highly cohesive and performed better in assigned tasks	
5	Choi et al. [28]		Exp	Stu (128)	The group with diverse pairs was more productive than similar and opposite pair groups	
6	Gilal et al. [29]		Exp	Stu (105)	By using personality, team role, and gender as predictors to formulate an effective team, RST proved to be the best data mining technique	
7	Poonam & Yasser [30]		Exp	Stu (80)	The performance of pairs working remotely was affected by personality traits	
8	Kamangar et al. [31]		Exp	Stu (92)	Testers with introvert personality types were more efficient in white box testing whereas extroverts were more productive in black-box testing	
9	Barroso et al. [32]		Exp	Pro (15)	Personality has a noteworthy impact on Depth of Inheritance Tree (DIT) and no notable influence was observed for Weighted Methods per Class (WMC) and Response For a Class (RFC) metrics	
10	Peeters et al. [21]		FFM/ Big Five	Exp	Pro & Stu (24-88 teams)	Homogeneous teams were better in solving structured tasks whereas heterogeneous teams were more useful in solving unstructured tasks
11	Walle & Hannay [34]			Sur	Pro (88)	Personality attributes contribute to the collaboration of a team.
12	Hannay et al. [35]			Reg	Pro (196)	Personality is a moderate predictor for pair performance
13	Chao and Atli [36]	Sur & Exp		Stu (58)	Pair success is not influenced by the variety in personality traits	
14	Feldt et al. [37]	Exp		Pro (47)	Software engineers having similar personality attributes have the same viewpoints	
15	Kanij et al. [38]	Exp		Stu (48)	Testers' performance had a positive correlation with the conscientiousness and extraversion personality attributes	
16	Salleh et al. [39]	Exp		Stu (453)	Conscientiousness and Neuroticism traits have no significant impact whereas Openness trait has a positive impact on teams' performance	
17	Acuña et al. [11]	Exp		Stu (136)	Extraversion has a positive correlation with software product quality and team climate factors affect the level of job satisfaction	
18	Lykourantzou et al. [40]	Exp		Pro (70)	Balanced teams' performance was much better with higher cooperation and lower conflicts	
19	M. Yilmaz et al. [13]	Sur		Pro (216)	Practitioners were found more extroverts and effective teams were observed to be emotionally more stable	
20	Barroso et al. [42]	Exp		Stu (25)	Agreeableness was found to be more significant in influencing the Coupling Between Objects (CBO) whereas neuroticism and extraversion were found to have a significant correlation with Depth of Inheritance Tree (DIT) and Coupling Between Objects (CBO) metrics of the CK metrics suite	
21	Amin et al. [43]	Sur		Pro (294)	The personality traits and knowledge collection behavior play a key role in shaping their intention to be creative.	
22	Yan et al. [44]	Reg		128	Adaptive and maladaptive personality traits are associated with passively sensed behavior	
23	Qamar & Malik [12]	Exp		Stu (135)	Teams with higher values of Team Homogeneity Index (THI) were more productive and produced better quality software during the implementation and testing phases	
24	Qamar & Malik [62]		Exp	Stu (215) & Pro (35)	A comparative analysis of Team Homogeneity Index and Weighted Team Homogeneity Index was performed and found that Weighted Team Homogeneity Index is more strongly correlated than Team Homogeneity Index for almost all of the teams	

Exp=Experiment, Sur=Survey, Reg=Regression, Pro=Professionals, Stu=Students

Personality Item Pool (IPIP) [48], etc.. IPIP is a web-based instrument developed by Goldberg and his fellows [48]. NEO-PI-R and NEO-FFI are proprietary tools while IPIP

is readily available in the public domain [48]. Personality psychologists developed IPIP with the help of international research collaboration. The purpose of this research

collaboration was to provide a reliable online automated inventory for personality profiling. Compared to any other paper-based personality instrument, an automated tool (i.e. IPIP) is a lot more effective [48]. Regarding the validity of the scales utilized to measure personality, the IPIP [48] is considered to have excellent reliability as compared to other well-known personality tools. The availability of IPIP in the public domain and its excellent reliability inspired us to use this as a personality measurement tool in our study.

#### IV. QUANTIFICATION OF TRAIT-WISE TEAM HOMOGENEITY

As indicated in Figure 1, the quantification of trait-wise team homogeneity is the first step of our research methodology followed by experiments conducted to assess the utility of the newly proposed metrics.

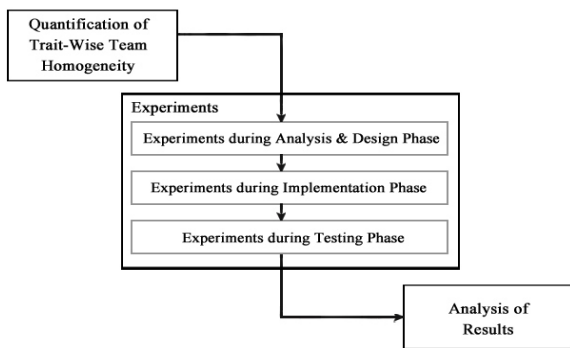


FIGURE 1. Research methodology.

The concept of trait-wise team homogeneity is defined as the calculation of Team Homogeneity Index (THI) [12] for each of the five personality traits of FFM (i.e. Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism). The process of quantification of trait-wise homogeneity consists of five steps. The first step is common for all five newly defined metrics (i.e. O-THI, C-THI, E-THI, A-THI, and N-THI). Steps 2 to 5 are different for each metric. The entire five-step process of calculation of one of these five new metrics (i.e. O-THI) is described below. The rest of the metrics are calculated in a similar fashion.

**Step 1:** Personality profiling of team members based on FFM is the first step in the process of quantification of trait-wise team homogeneity. An International Personality Item Pool (IPIP) based 50-item personality test is used for this purpose [48]. A personality score (measured on an interval scale) for each of the five personality traits is determined for every team member. Next, the personality score is confined to a specific range (i.e. 0-1) using the Min-Max normalization technique [49] as the distribution of personality scores was unknown.

Figure 2 shows an example of openness heterogeneity calculation for a sample five-member team. Four red arrow-lines going out from M1 to all other members indicate how M1 is different from the rest of the team members (i.e. M2, M3, M4, and M5) with respect to the openness trait. Three green arrow-lines go out from M2 to the rest of the team members

except M1 (whose heterogeneity with M2 has already been calculated). The same process is repeated for M3 (two orange arrow-lines go out to M4 and M5) and M4 (one blue arrow-line goes out to M5). No further processing is required for the last member (i.e. M5) as this member has already been compared with all the other team members. As a result of this step, a matrix (see Figure 3 – Step 2) is obtained. Lower-diagonal values are used to avoid redundancy.

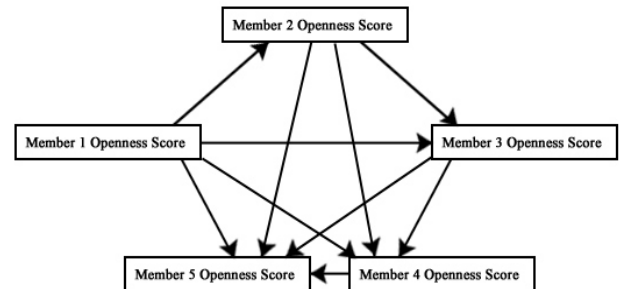


FIGURE 2. Heterogeneity of openness personality trait for a five-member team.

**Step 2:** During this step, the calculation of heterogeneity between the score of one team member with every other team member for the openness trait (as demonstrated in Figure 3) takes place using Equation (1). This step will result in the heterogeneity scores for all possible non-duplicated member-pairs (e.g. M1 & M2, M1 & M3, M1 & M4, M1 & M5, M2 & M3, M2 & M4, M2 & M5, M3 & M4, etc.)

$$H = |x - y| \tag{1}$$

where

H = heterogeneity between x and y

x = score of the one team member for openness trait

y = score of another team member for openness trait

**Step 3:** Calculation of Openness Mean Heterogeneity (OMH) is the third step of this process. The OMH is calculated (using Equation (2)) by taking the average of all member-pairs' heterogeneity scores (calculated in step 2).

$$OMH = \frac{1}{m} \left( \sum_{k=1}^{k=m} H_k \right) \tag{2}$$

where

OMH = Openness Mean Heterogeneity

H<sub>k</sub> = heterogeneity of kth member-pairs

m = total number of member-pairs (n(n-1)/2 => where n=number of team members)

**Step 4:** In this step, Openness Mean Absolute Error (OMAE) is computed (using Equation (3)) using the sum of the absolute difference between OMH (calculated in step 3) and H (calculated in step 2) for all member-pairs and dividing this sum by the total number of member-pairs (e.g. 10 for the five-member team shown in Figures 3 and 4).

$$OMAE = \frac{1}{m} \sum_{k=1}^m |OMH - H_k| \tag{3}$$

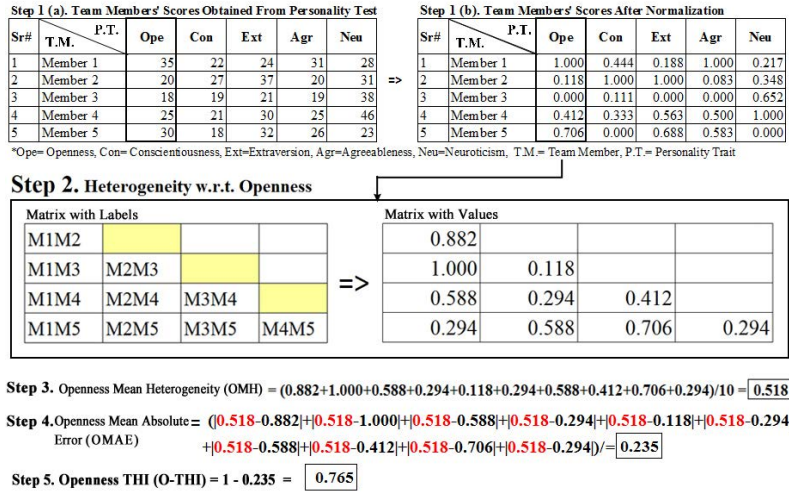


FIGURE 3. Worked-out example of O-THI calculation.

where

OMAE = Openness Mean Absolute Error

OMH = Openness Mean Heterogeneity

Hk = heterogeneity of the kth member-pair

m = total number of member-pairs  $(n(n-1)/2) \Rightarrow$  where n=number of team members)

**Step 5:** Lastly, O-THI (which captures the similarity/homogeneity in a team with respect to the openness trait) is calculated by subtracting OMAE (calculated in step 4) from 1 using Equation (4). THI for any trait lies in the range of 0 to 1 where 0 means that the team is completely heterogeneous while 1 means that the team is completely homogeneous.

$$O-THI = 1 - OMSE \quad (4)$$

A worked-out example of the O-THI calculation using the above-mentioned five-step process is shown in Figure 4. First, the personality scores of team members were captured and normalized. Then, in step 2, heterogeneity of one team member with the rest of the team members was calculated for the openness trait (using Equation (1)). In the next step, Openness Mean Heterogeneity (OMH) was calculated using Equation (2) followed by the calculation of openness mean square error (OMSE) using Equation (3). Finally, O-THI was computed using Equation (4).

## V. EXPERIMENTATION

The goals of our experiments were defined by using the goal-question-metric (GQM) framework [50]. The concept of GQM was defined by Basili and Rombach in 1988 [51] to provide a systematic approach to define goals of an empirical study (i.e. experiments) and determine which metrics are appropriate to achieve a specific goal. Table 2 provides the detailed mapping to the GQM framework while the mapping to the structured GQM template is given below:

**Analyze** – trait-wise team homogeneity (i.e. O-THI, C-THI, E-THI, A-THI, and N-THI)

**For the purpose of** – evaluation

TABLE 2. Mapping to GQM framework.

Goal(s)	Question(s)	Metric(s)
Evaluation of the impact of trait-wise team homogeneity (i.e. O-THI, C-THI, E-THI, A-THI, and N-THI) on team productivity and software quality	Does homogeneity/similarity in personality traits of team members affect software quality?	<b>Metrics for software quality</b> For analysis and design phase: <i>Understandability of Models (Und)</i> <i>Correctness of Models (Cor)</i> <i>Layout of Models (Lay)</i> <i>Appropriately Used Relationships in Models (Rel)</i>
	Does homogeneity/similarity in personality traits of team members affect team productivity?	For implementation phase: <i>Weighted Sum of Bugs (WSB)</i> <i>Cyclomatic Complexity (CC)</i> <i>Defect Density (DD)</i> <i>Maintainability Index (MI)</i> For testing phase: <i>Defects Uncovered (DU)</i> <i>Requirements Coverage (RC)</i> <i>Test Case Conformity (TCC)</i>
		<b>Metrics for team productivity</b> For analysis and design phase: <i>Models' Completeness per Person Hour of Effort (MC/E)</i> For implementation phase: <i>Project Completeness per Person Hour of Effort (PC/E)</i> For testing phase: <i>No. of Features Tested per Person Hour of Effort (FT/E)</i>

**With respect to** – team productivity and software quality  
**From the point of view of** – software practitioners and researchers

**In the context of** – software industry professionals and undergraduate students (enrolled in “Software Engineering”,

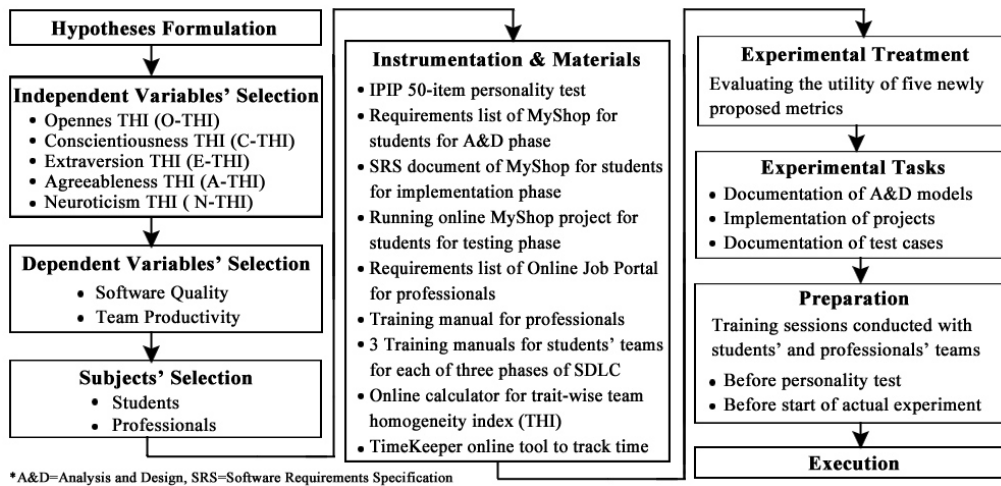


FIGURE 4. Overview of experimental design.

“Web Engineering”, and “Software Testing” courses). Professionals performed experiments in their respective software houses while students performed experiments in university labs.

An overview of our experimental design [52] for the three different phases of the SDLC is given in Figure 4 and the details are provided in subsequent sections.

#### A. HYPOTHESES/RESEARCH QUESTIONS FORMULATION

During these experiments, the following hypotheses were tested for the three different phases of SDLC i.e. analysis and design, implementation, and testing. There are five independent (discussed in the next section) and two main dependent variables (discussed in Section 5.3) in our experiments. To find the correlation between them there are  $5 \times 2 = 10$  possible combinations. We proposed null and alternate hypotheses for each combination.

H\_QO0: O-THI values have no/negative impact on software quality.

H\_QO1: O-THI values have a positive impact on software quality.

H\_QC0: C-THI values have no/negative impact on software quality.

H\_QC1: C-THI values have a positive impact on software quality.

H\_QE0: E-THI values have no/negative impact on software quality.

H\_QE1: E-THI values have a positive impact on software quality.

H\_QA0: A-THI values have no/ negative impact on software quality.

H\_QA1: A-THI values have a positive impact on software quality.

H\_QN0: N-THI values have no/positive impact on software quality.

H\_QN1: N-THI values have a negative impact on software quality.

H\_PO0: O-THI values have no/negative impact on team productivity.

H\_PO1: O-THI values have a positive impact on team productivity.

H\_PC0: C-THI values have no/negative impact on team productivity.

H\_PC1: C-THI values have a positive impact on team productivity.

H\_PE0: E-THI values have no/negative impact on team productivity.

H\_PE1: E-THI values have a positive impact on team productivity.

H\_PA0: A-THI values have no/negative impact on team productivity.

H\_PA1: A-THI values have a positive impact on team productivity.

H\_PN0: N-THI values have no/positive impact on team productivity.

H\_PN1: N-THI values have a negative impact on team productivity.

#### B. INDEPENDENT VARIABLES' SELECTION

The concept of trait-wise team homogeneity was quantified using five different metrics. These metrics served as the five independent variables in this research. The definitions of these five metrics are given below:

**Openness THI (O-THI)** – degree of similarity/homogeneity in openness level between team members

**Conscientiousness THI (C-THI)** – degree of similarity/homogeneity in conscientiousness level between team members

**Extraversion THI (E-THI)** – degree of similarity/homogeneity in extraversion level between team members

**Agreeableness THI (A-THI)** – degree of similarity/homogeneity in agreeableness level between team members

**Neuroticism THI (N-THI)** – degree of similarity/homogeneity in neuroticism level between team members



### C. DEPENDENT VARIABLES' SELECTION

Team productivity and software quality were our two main dependent/response variables which were calculated using different metrics. Table 3 describes the details of our dependent variables, metrics used to measure these variables, and formulas for these metrics for the three different phases of SDLC.

For the analysis and design phase, software quality was measured using four metrics (i.e. understandability, correctness, layout, and appropriately used relationships). Each of these metrics was calculated by taking the average of the rating scores of all the models (i.e. class, entity-relationship, data-flow, sequence, and activity diagrams). These scores were awarded on a scale of 1 to 10. 1 represents the least score for any selected metric and 10 represents the highest score for that specific metric. Team productivity was calculated by taking the ratio between models' completeness and effort taken in person-hours (PH) to design the models. Table 4 provides the details of the analysis and design phase metrics along with their components used to calculate metrics' scores.

For the implementation phase, software quality was measured in terms of weighted sum of bugs (WSB), cyclomatic complexity (CC), defects density (DD), and maintainability index (MI). To measure WSB, first, we assigned different weights to different levels of severity of bugs. For example, a weight of 5 was assigned to a bug with critical severity, a weight of 4 was assigned to a bug with major severity, and so on. Later, the sum of those weighted bugs was calculated. To calculate DD, the WSB was divided by lines of code (LOC). Cyclomatic Complexity (CC) [53] determines how hard it is to understand a project whereas the Maintainability Index (MI) [54] deals with the ease of making changes in the project. CC and MI were determined using the PHP Metric [55] tool. As far as team productivity of this phase is concerned, it was calculated by taking the ratio between project completeness (PC) and effort in PH. PC was calculated by looking at percentage completeness of requirements (e.g. 100% implemented requirements, 50% implemented requirements, etc.).

For the testing phase, software quality was measured using defects uncovered (DU), requirements coverage (RC), and test case conformity (TCC). DU was simply measured by looking at the number of failed test cases. RC was calculated by using the ratio of tested requirements and total requirements specified while TCC was calculated by using the ratio of correct test case attributes and total test case attributes documented by teams. The productivity of testing teams was determined by taking the ratio of the total number of tested features and the effort taken in person-hours (PH) to document the test cases corresponding to these features.

### D. SUBJECTS' SELECTION

These experiments were carried out in academic as well as industrial environments. Therefore, both students and industry professionals were selected to take part in our experiments. Table 5 provides the complete details of the selected subjects.

Thirty-five professionals (33 males and 2 females) and 215 students (197 males and 18 females) participated in these experiments. The recruitment of professionals was based on an agreement with their associated software houses that they will own the copyrights of the developed projects and we will use projects' data only for research purposes. The selected professionals worked on each of the three phases of SDLC i.e. analysis and design, implementation, and testing. In the case of students, different groups of students were selected for each of the three different phases. A group consisting of fifty students enrolled in the "Software Engineering" course was chosen for the analysis and design phase. Ninety students from the "Web Engineering" course and seventy-five students from the "Software Testing" course took part in the implementation and testing phases, respectively. Both types of subjects (i.e. students and professionals) were neither aware of the actual hypotheses of this research nor were they informed about what we intended to investigate in these experiments.

All of the selected subjects (students and professionals) were divided into teams. Each team consisted of five members. In the case of students, to keep all other factors (other than personality traits) constant, teams were created using the following strategy:

- Students were divided into the following three groups based on their CGPA

Group A: 3.00 – 4.00

Group B: 2.50 – 2.99

Group C: 2.00 – 2.49

Each student team was formulated by choosing one member from "Group A", two members from each of "Group B" and "Group C". This selection from different groups was done randomly. The average CGPA of each team was in the range of 2.50-2.80.

- Mean grade of each team in programming subjects was not less than 'B'
- A maximum of one female member was allowed in each team
- No more than one student with previous work experience (e.g. freelancing, internship, etc.) was allowed in one team

Similarly, in the case of professionals, to keep all other factors constant, the teams were formulated using the following strategy:

- The minimum qualification of each professional was a bachelor's degree in computing
- The average work experience of each team was in the range of 3-4 years
- The average age of each team was in the range of 24-26 years
- A maximum of one female member was allowed in a single team

### E. INSTRUMENTATION AND MATERIALS

There were a total of eleven instruments and materials used in these experiments: i) IPIP 50-item personality test for

**TABLE 3. Details of dependent variables.**

Phase	Dependent Variable	Metric	Formulas/Equations Used for Calculating Metric
Analysis and Design	Software Quality	Understandability of Models (i.e. Class, ER, DF, Sequence, and Activity diagrams) (Und)	Average of all the models' understandability scores (Score of each model was awarded on a scale of 1-10 where 1=least understandable and 10=most understandable)
		Correctness of Models (Cor)	Average of all the models' correctness scores (Score of each model was awarded on a scale of 1-10 where 1=least correct and 10=most correct)
		Layout of Models (Lay)	Average of all the models' layout scores (Score of each model was awarded on a scale of 1-10 where 1=most complex layout and 10=least complex layout)
		Appropriately Used Relationships in Models (Rel)	Average of all the relationships used at appropriate places in each model (Relationships include generalization, aggregation, association, and composition)
	Team Productivity	Productivity (Prod)	$Models' \text{ Completeness } (MC) = \text{average of percentage completeness of each model}$  $Effort(E) = \text{person hours taken to design models}$  $Prod = MC/E$
Implementation	Software Quality	Weighted Sum of Bugs (WSB)	$WSB = 5(\# \text{ of critical } B) + 4(\# \text{ of major } B) + 3(\# \text{ of moderate } B) + 2(\# \text{ of minor } B) + 1(\# \text{ of low } B)$  B=bugs identified
		Cyclomatic Complexity (CC)	$CC = E - N + 2P$  E= no. of edges N= no. of nodes P= no. of connected components
		Defect Density (DD)	$DD = WSB / \text{Lines of Code } (LOC)$
		Maintainability Index (MI)	$MI = 171 - 5.2 * \ln(HV) - 0.23 * (CC) - 16.2 * \ln(LOC)$  HV= Halstead volume
	Team Productivity	Productivity (Prod)	$Project \text{ Completeness } (PC) = (100(IR1) + 75(IR2) + 50(IR3) + 25(IR4) + 0(IR5)) / 100$  IR1= 100% implemented requirements IR2= 75% implemented requirements IR3=50% implemented requirements IR4=25% implemented requirements IR5= not implemented requirements  $Effort(E) = \text{person hours taken to implement the project}$  $Prod = PC/E$
Testing	Software Quality	Defects Uncovered (DU)	$DU = \text{Failed Test Cases}$
		Requirements Coverage (RC)	$RC(\%) = (\text{Tested Requirements} / \text{Total Requirements Specified}) * 100$
		Test Case Conformity (TCC)	$TCC(\%) = (CTA / TTA) * 100$  CTA = correct test-case attributes TTA = total test-case attributes
	Team Productivity	Productivity (Prod)	$Tested \text{ Features } (TF) = \text{Total no. of features tested by teams}$  $Effort(E) = \text{person hours taken to document test cases}$  $Prod = TF/E$

\* DF=Data-Flow, ER=Entity Relationship

**TABLE 4.** Analysis and design phase metrics and their components.

Metric	Components		
Understandability	Easy to Read	Relationships Names	Extra Information
	Complexity	No of Classes, Attributes, & Operations	Class, Attributes, & Operations Names
Correctness	Correctness of Classes	Correctness of Operations	Correctness of Data Flow
	Correctness of Attributes	Correctness of Sequences	Correctness of Layout
	Correctness of Entities	Conformance to the Standards	Correctness of Relationships
Layout	Good Class Name	Classes Hierarchy, Alignment	Neat or Chaotic Structure
	Classes with Similar Size	Number of Compositions	Distance between Classes
Use of Relationships at Appropriate Places	Number of Aggregations	Number of Generalizations	Number of Compositions
	Number of Associations		
Completeness	Model Abstraction	Missing Classes, Attributes, & Operations	Strange Relationships
	Functionality	Functions Parameters	Data Types

**TABLE 5.** Subjects characteristics.

Attributes	Undergraduate Students			Professionals
	A&D Phase	Imp. Phase	Testing Phase	A&D, Imp., and Testing Phases
Enrolled Semester	7 <sup>th</sup>	6 <sup>th</sup>	8 <sup>th</sup>	N/A
Relevant Registered Course	Software Engineering	Web Engineering	Software Testing	N/A
No. of Male Participants	46	81	70	33
No. of Female Participants	4	9	5	2
No. of Total Participants	50	90	75	35
No. of Teams	10	18	15	7
Members in Each Team	5	5	5	5
Task Duration	2 Weeks	4 Weeks	2 Days	8 Weeks
Overall Experiment Duration (includes preparation and training)	6 Weeks	8 Weeks	4 Weeks	12 Weeks
*A&D=Analysis and Design, Imp.=Implementation				

personality profiling, ii) Requirements list of MyShop project for students' teams for analysis and design phase, iii) Software Requirements Specification (SRS) document of MyShop project for students' teams for the implementation phase, iv) Running online MyShop project for students' teams for the testing phase, v) Requirements list for Online Job Portal project for professionals' teams for all three phases, vi) Training manual for professionals for all three phases, vii) Training manual for students' teams for the analysis and design phase, viii) Training manual for students' teams for the implementation phase, ix) Training manual for students' teams for the testing phase, x) Online calculator [56] for trait-wise team homogeneity, and xi) Time-Keeper online tool [57] to log time. All of the raw data generated during these experiments was made available on the web [56], [57].

Myshop project (for students) was composed of 12 modules and Online Job Portal project (for professionals) was composed of 19 modules. There were a total of 51 requirements in SRS document of MyShop project whereas 71 requirements were part of Online Job Portal's requirements

list. MyShop project was written using 547,397 lines of code and Online Job Portal was written using 987,694 lines of code. Both projects were developed using PHP programming language for backend and HTML/CSS/JS for front end development.

The IPIP 50-item personality test (10 items belong to each personality trait) was conducted before experiments for personality profiling (i.e. getting scores for the five traits of FFM) of the participants. These scores were further used for calculating the five metrics (our independent variables) introduced for quantification of trait-wise team homogeneity (Section 3). The MyShop project was selected for students' teams and the Online Job Portal project was selected for professionals' teams as experimental objects. Training manuals were used during training sessions conducted with subjects (students and professionals) before the start of the personality tests and actual experiments. An online calculator (developed by the first author as part of her PhD research) was used to automatically calculate trait-wise team homogeneity (i.e. O-THI, C-THI, E-THI, A-THI, and N-THI).

TABLE 6. Training sessions’ details.

Session	Duration	Group	Participants
<b>Students (Analysis &amp; Design Phase)</b>			
S1 – Before Personality Profiling	½ hour	Group 1	50
S2 – Before Experiment	3 hours		
<b>Students (Implementation Phase)</b>			
S3 & S4 – Before Personality Profiling	½ hour	Group 1 & Group 2	45
S5 & S6 – Before Experiment	3 hours	Group 1 & Group 2	
<b>Students (Testing Phase)</b>			
S7 – Before Personality Profiling	½ hour	Group 1	75
S8 – Before Experiment	3 hours		
<b>Professionals (Analysis &amp; Design, Implementation, and Testing Phase)</b>			
S9 – Before Personality Profiling	½ hour	Group 1	35
S10 – Before Experiment	3 hours		

**F. EXPERIMENTAL TREATMENT**

We designed one-factor “between subjects” experiments i.e. the comparison was between different subjects to evaluate the impact of one independent variable (each of the five proposed metrics i.e. O-THI, C-THI, E-THI, A-THI, and N-THI) on team productivity and software quality (dependent variables) in each treatment. Therefore, there were a total of five treatments in our experiments. All the experiments followed the same experimental procedure.

**G. EXPERIMENTAL TASKS**

The task for students’ teams during the analysis and design phase was to design different models (i.e. class, entity-relationship (ER), data-flow (DF), sequence, and activity diagrams) for MyShop project. For the implementation phase, students’ teams were asked to write the complete code for the MyShop project. For the testing phase, the students’ teams were given the task of black-box testing of the running MyShop project using the equivalence class partitioning (ECP) approach. The professionals’ teams also performed the same tasks as students’ teams but on a relatively bigger and more complex project i.e. Online Job Portal.

**H. PREPARATION**

To successfully conduct the experiments, we had to prepare the subjects (by providing guidelines, conducting training sessions, etc.), instrumentation, and materials (e.g. documents, projects, training manuals, etc.), and data collection mechanisms (i.e. online personality test and calculator for trait-wise homogeneity).

To prepare the subjects for the experiments, a total of ten training sessions were carried out with different groups of subjects. Table 6 provides the details of these training sessions. These sessions were carried out at two points i.e. 1/2 hour session before the personality profiling and 3-hours session before the experiment) for each group of participants.

For analysis and design and testing phases involving students, two training sessions were conducted in each phase - one 1/2 hour session before the personality profiling (S1 and S7) and one 3-hours session (S2 and S8) before starting the experiment. Similarly, two sessions (one 1/2 hour session (S9) and one 3-hours session (S10)) were conducted with professionals. Students’ training sessions were conducted during their weekly lab sessions (3-hours duration) whereas professionals’ training sessions were conducted online using the “Skype” tool.

Only in the implementation phase involving students, due to a larger number of participants, subjects were divided into two groups to maintain the quality of training sessions. Therefore, four training sessions were conducted. Two 1/2 hour sessions (S3 and S4) were carried out before the personality profiling and two 3-hours training sessions (S5 and S6) were conducted before the experiment.

The agenda of the students’ training session conducted before the actual experiment during the analysis and design phase was as follows:

- A comprehensive introduction of MyShop project’s requirements list
- Explanation of the five analysis and design models (class, entity-relationship, data flow, activity, and sequence diagrams) students were expected to produce
- Explanation of the experimental process during the analysis and design phase
- Introduction and demonstration of online tool to record time (i.e. “TimeKeeper” [57])
- Analysis and design guidelines:
  - All the models must be created using “Visio” tool
  - Out of five team members, one will play the role of team lead
  - Teams will have two weeks to submit the task

The agenda of the students’ training sessions conducted before the actual experiment during the implementation phase was as follows:

- A comprehensive introduction of SRS document of MyShop project
- Explanation of the experimental process during the implementation phase
- Implementation guidelines:
  - Project architecture must be based on the Model View Controller (MVC) pattern
  - The view/front-end part of the app will be created using “HTML4/HTML5” and “Bootstrap” framework will be used for styling
  - For client-end scripting, teams will use “JavaScript” programming language
  - For back-end/server-end programming, teams will use “PHP” programming language
  - For database management, teams will use “MySQL” tool
  - Teams will use the “Waterfall” software development process
  - One team member will lead the team and the rest will work as team members
  - Maximum duration for the project implementation will be four weeks
- Introduction and demonstration of online tool to record time (i.e. “TimeKeeper” [57])

The agenda of the students’ training session conducted before the actual experiment during the testing phase was as follows:

- Demonstration of running MyShop project
- An explanation of SRS document of MyShop project
- Introduction to the test case template to document test cases
- Explanation of the experimental process during the testing phase
- Testing guidelines:
  - Teams will do “Black-Box/Functional” testing
  - Teams will follow the given template [58]
  - Test cases will be documented using the “Equivalence Class Partitioning” technique
  - Test cases will be based on the given SRS document
  - One member will lead the team and the others will perform as team members
  - Teams will have two working days to finish their task
- Introduction and demonstration of online tool to record time (i.e. “TimeKeeper” [57])

The agenda of the online training session (for professionals) conducted before the actual experiment was as follows:

- A comprehensive introduction of the requirements list of the Online Job Portal project
- Explanation of the experimental process during three phases of SDLC (i.e. analysis and design, implementation, and testing)
- Guidelines for analysis and design, implementation, and testing:
  - The project will have three milestones (one at the end of each phase) and teams will have to submit

three deliverables: i) document containing analysis and design models, ii) complete code, and iii) test case document

- Teams will have two weeks for the first deliverable, five weeks for the second deliverable, and one week for the third deliverable
- One team member will play the role of team lead and rest will serve as team members
- All other guidelines related to techniques and tools will be the same as for students’ teams (e.g. “Visio” tool for design, MVC, HTML, Bootstrap, MySQL, Waterfall process, Equivalence Class Partitioning, compliance to given template for testing, etc.)
- Introduction and demonstration of online tool to record time (i.e. “TimeKeeper” [57])

## I. EXECUTION

After successful completion of training sessions, MyShop’s requirements list was provided to students’ teams who were part of the experiment conducted during the analysis and design phase. Similarly, MyShop’s SRS document was handed over to the teams who participated in the experiment carried out during the implementation phase. Access of a fully developed online MyShop project was given to the teams working for the testing phase. The requirements list of the Online Job Portal project was provided to the professionals’ teams for analysis & design, implementation, and testing purposes.

In the case of students, for the analysis and design phase, teams were given two weeks to complete their analysis and design models. Four weeks were given for implementation and two working days were given for the testing task. Professionals were asked to submit the first deliverable (analysis and design models) after two weeks, second deliverable (implemented project) after five weeks, and the last deliverable (test case document) after one week. Each team member was given access to the “TimeKeeper” tool at the beginning of the projects’ execution. Dropout students were not made a part of any team.

## VI. RESULTS AND DISCUSSION

This section provides the results of the experiments we have conducted in three different phases of SDLC i.e. analysis and design, implementation, and testing. The results were evaluated based on the selected criteria (described in Table 3).

### A. ANALYSIS AND DESIGN PHASE

Table 7 shows the bivariate (Pearson) correlation [61] between the dependent and independent variables for both types of subjects (i.e. students and professionals). Correlation (r) values range from -1 to +1. An r value of -1 represents a perfect negative correlation, +1 represents a perfect positive correlation, and 0 represents no correlation. Absolute r values between 0.1 and 0.3 indicate a weak correlation, values between 0.3 and 0.7 represent a moderate correlation, and values between 0.7 and 1 indicate a strong correlation [61].

**TABLE 7. Correlation results of analysis and design phase.**

Subject Type	Trait-Wise Team Homogeneity	correlation (r)				
		Software Quality				Team Prod
		Und	Cor	Lay	Rel	
Students	O-THI	<u>0.77</u>	<u>0.71</u>	0.53	0.53	0.43
	C-THI	<u>0.71</u>	<u>0.75</u>	0.64	0.65	0.65
	E-THI	0.48	0.35	0.43	0.21	0.20
	A-THI	<u>0.79</u>	0.69	0.47	0.45	0.48
	N-THI	0.46	0.32	0.33	0.28	0.30
Professionals	O-THI	0.55	0.08	0.35	0.33	0.00
	C-THI	<u>0.79</u>	0.42	0.60	0.60	0.50
	E-THI	0.50	0.55	0.57	0.55	0.22
	A-THI	0.62	0.46	<u>0.71</u>	<u>0.70</u>	0.17
	N-THI	-0.15	-0.16	-0.03	-0.05	-0.53

Und=Understandability, Cor=Correctness, Lay=Layout, Rel=Appropriately Used Relationships, Prod=Productivity

**TABLE 8. Correlation results of implementation phase.**

Subject Type	Trait-Wise Team Homogeneity	correlation (r)				
		Software Quality				Team Prod
		WSB	CC	DD	MI	
Students	O-THI	-0.52	-0.37	-0.37	-0.02	0.64
	C-THI	-0.52	-0.40	<u>-0.76</u>	0.39	0.51
	E-THI	-0.52	-0.55	-0.52	0.09	0.65
	A-THI	-0.43	-0.36	-0.20	-0.03	0.41
	N-THI	0.08	-0.12	0.01	0.05	-0.06
Professionals	O-THI	-0.09	-0.54	-0.29	<u>0.79</u>	0.65
	C-THI	-0.48	<u>-0.80</u>	-0.68	<u>0.82</u>	0.51
	E-THI	-0.34	-0.36	-0.56	0.48	0.33
	A-THI	-0.45	0.10	-0.37	-0.18	0.16
	N-THI	0.08	0.64	0.26	<u>-0.80</u>	-0.37

WSB=Weighted Sum of Bugs, CC=Cyclomatic Complexity, DD=Defect Density, MI=Maintainability Index, Prod=Productivity

Table 7 highlights the strong correlation values as “bold-underlined” whereas the moderate correlation values are presented in simple “bold”. Weak correlation values, on the other hand, are presented in regular font. It is evident from these correlation results that O-THI (for students only), C-THI, and A-THI have a moderate to strong positive correlation with almost all attributes of software quality. N-THI, on the other hand, has a weak negative correlation with software quality for professionals. Team productivity has a moderate positive correlation with O-THI, C-THI, A-THI, and N-THI in the case of students. For professionals, it has a moderate positive correlation with C-THI and a moderate negative correlation with N-THI.

These correlation results describe that, apart from N-THI (for professionals), higher values of all other trait-wise team homogeneity indices are associated with improved software quality. These results support our hypotheses H\_QO1,

H\_QC1, H\_QE1, H\_QA1, H\_QN1 (for professionals only) H\_PO1 (for students only), H\_PC1, H\_PE1, H\_PA1 (for students only), and H\_PN1 (for professionals only).

**B. IMPLEMENTATION PHASE**

Table 8 presents the summary of the correlation between dependent and independent variables for the implementation phase. The first two columns contain the subject type and independent variables. Columns 3-7 provide the correlation between trait-wise team homogeneity and software quality and team productivity.

In the case of students, a low to moderate negative correlation exists between O-THI and A-THI and all attributes of software quality while a moderate to high negative correlation exists between C-THI and E-THI and the first three attributes (i.e. WSB, CC, and DD) of software quality. N-THI has a very weak correlation with all attributes of software quality. These

TABLE 9. Correlation results of testing phase.

Subject Type	Trait-Wise Team Homogeneity	correlation (r)			
		Software Quality			Team Prod
		DU	RC	TCC	
Students	O-THI	0.27	0.27	-0.18	-0.03
	C-THI	0.30	<b>0.45</b>	0.22	<b>0.42</b>
	E-THI	0.10	-0.11	-0.08	-0.17
	A-THI	<b>0.67</b>	<b>0.51</b>	<b>0.32</b>	<b>0.44</b>
	N-THI	0.29	<b>0.51</b>	0.04	<b>0.61</b>
Professionals	O-THI	<b>0.69</b>	<b>0.31</b>	0.28	<b>0.36</b>
	C-THI	<b>0.41</b>	<b>0.49</b>	<b>0.51</b>	<b>0.62</b>
	E-THI	0.20	<b>0.60</b>	<b>0.49</b>	<b>0.56</b>
	A-THI	0.26	<b>0.55</b>	<b>0.58</b>	0.27
	N-THI	<b>-0.46</b>	0.26	-0.09	<b>-0.43</b>

DU=Defects Uncovered, RC=Requirements Coverage, TCC=Test Case Conformity, Prod=Productivity

correlation results indicate that, apart from N-THI, higher values of all other trait-wise team homogeneity indices are associated with improved software quality. As far as team productivity is concerned, a moderate positive correlation exists with all trait-wise THIs except N-THI (which has a moderate negative correlation) implying that higher values of O-THI, C-THI, E-THI, and A-THI are associated with improved productivity. The opposite is true for N-THI.

For professional subjects, C-THI and E-THI have a moderate to strong negative correlation with WSB, CC, and DD while a strong positive correlation exists with MI. This implies that higher values of these two trait-wise THIs are associated with better quality. Correlation results of O-THI imply the same conclusion except for WSB and DD. A-THI and N-THI do not follow the same pattern as O-THI, C-THI, and E-THI. They both have a negative correlation with MI and a positive correlation with CC. As far as team productivity is concerned, the results are similar to the ones for student subjects i.e. higher values of O-THI, C-THI, E-THI, and A-THI are associated with improved productivity while the opposite is true for N-THI.

Even though a number of exceptions exist, the above results indicate that, by and large, higher values of O-THI, C-THI, E-THI, and A-THI have a positive impact on software quality while the opposite is true for N-THI. These results, therefore, support our hypotheses H\_QO1 (except MI for students), H\_QC1, H\_QE1, H\_QA1 (except CC for professionals and MI for both students and professionals), and H\_QN1 (except CC and MI for students) during the implementation phase. All hypotheses related to productivity (i.e. H\_PO1, H\_PC1, H\_PE1, H\_PA1, and H\_PN1) are also supported by these results of the implementation phase.

C. TESTING PHASE

Table 9 presents the results of the correlation between dependent (software quality and team productivity) and

independent variables (O-THI, C-THI, E-THI, A-THI, and N-THI) for the testing phase. The information about the subject type and independent variables are provided in the first two columns. Columns 3-5 contain the correlation values between independent variables and software quality whereas the last column contains the correlation values for team productivity.

In the case of students, O-THI has a weak positive correlation with two quality attributes (i.e. DU and RC) and E-THI has a weak positive correlation with one quality attribute (i.e. DU). C-THI, A-THI, and N-THI have a low to moderate positive correlation with all three attributes of software quality. The same three trait-wise THIs have a moderate positive correlation with team productivity. These results imply that higher values of these three THIs are associated with better software quality and improved productivity. In the case of professionals, results show that, higher values of the first four trait-wise team homogeneity indices (i.e. O-THI, C-THI, E-THI, and A-THI) are associated with improved software quality and better productivity. Higher values of N-THI, however, are associated with a negative impact on quality and team productivity.

These results support hypotheses H\_QO1 (except TCC for students), H\_QC1, H\_QE1 (except RC and TCC for students), and H\_QA1. Hypothesis H\_QN1 is not supported for students and partially supported (for DU and TCC) for professionals. As far as team productivity is concerned, hypotheses H\_PC1 and H\_PA1 are supported for both students and professionals whereas hypotheses H\_PO1, H\_PE1, and H\_PN1 are supported only for professionals.

The results of this study show that, in the case of professionals, team homogeneity for Openness (O-THI), Conscientiousness (C-THI), and Extraversion (E-THI) traits has a stronger correlation with software quality and team productivity during the implementation phase. These findings are supported by [11], [12], and [13]. A possible reason behind

TABLE 10. Trait-wise comparison with previous studies.

Sr#	Author(s) (Year)	Ope	Con	Ext	Agr	Neu
1	Peeters et al. [21]		+ve		+ve	-ve
2	Salleh et al. [39]	+ve	-ve			-ve
3	Acuña et al. (2015)			+ve	+ve	
4	Yilmaz et al. [13]		+ve	+ve	+ve	-ve
5	This Study	+ve	+ve	+ve	+ve	-ve

Agr=Agreeableness, Con=Conscientiousness, Ext=Extraversion, Neu=Neuroticism, Ope=Openness, +ve=Positive Correlation, -ve=Negative Correlation

this could be that more communication and coordination is needed during the implementation phase and professionals having Openness and Extraversion traits are in a better position to coordinate and communicate with each other. The homogeneity in Conscientiousness (C-THI) and Extraversion (E-THI) traits were found to be strongly related to software quality and team productivity for the testing phase. These findings are partially supported by [31] and [38]. The traits that were found to be useful for implementation phase and testing phase are also useful for the analysis and design phase. In the case of students, a strong correlation was observed between O-THI, C-THI, and E-THI and software quality during the implementation phase and between C-THI and A-THI and software quality and team productivity during the testing phase. During the analysis and design phase, C-THI has a stronger correlation with software quality. A possible reason behind this relationship could be that more concentration and Conscientiousness is needed during the analysis and design process.

Table 10 summarizes the trait-wise comparison between the findings of our study and the results of previous studies analyzing the impact of personality traits on team performance. A “+ve” sign indicates a positive correlation between a personality trait and team performance whereas a “-ve” sign represents a negative correlation between a personality trait and team performance. As is clear from this comparison, by and large, the findings of our study match those of past similar studies.

**D. HYPOTHESES TESTING**

We used one-way Analysis of Variance (ANOVA) to test the hypotheses related to team productivity and Multivariate Analysis of Variance (MANOVA) to test the hypotheses related to software quality [49]. MANOVA is the same as ANOVA with two or more response variables. In our experiments, we have used multiple attributes of software quality (main response variable). Therefore, MANOVA helped us to conduct hypotheses testing (with lesser number of tests) by considering all the attributes of software quality (e.g. understandability, correctness, layout, and appropriately used relationship for analysis and design phase) simultaneously. In total, we have conducted 15 tests for software quality. We used the “General Linear Model” program of the SPSS

tool [60] to conduct MANOVA and ANOVA tests. A threshold p-value of 0.05 or above was used.

Table 11 shows the results of the MANOVA tests (based on “Roy’s Largest Root” model) for software quality. These results indicate that, for the analysis and design phase, we can reject all null hypotheses related to software quality except H\_QE0. These findings indicate that homogeneity in Extraversion personality trait is not necessary for the analysis and design phase. Members having homogeneity in Openness, Conscientiousness, and Agreeableness, and lower Neuroticism are good for the analysis and design process. Similarly, in the case of the implementation phase, all null hypotheses related to software quality can be rejected except H\_QA0. These findings reveal that teams having higher values of A-THI are suitable for project implementation. For the testing phase, on the other hand, only two (out of five) null hypotheses (i.e. H\_QE0 and H\_QN0) can be rejected. This shows that homogeneity for Extraversion and Neuroticism does not contribute much for the testing phase.

Table 12 presents the results of the ANOVA tests conducted to evaluate the statistical significance of our results for team productivity. It is evident from these results that the majority (i.e. 11 out of 15) of the null hypotheses related to team productivity cannot be rejected. Only H\_PA0 for the analysis and design phase, H\_PEO for the implementation phase, and H\_PCO and H\_PNO for the testing phase can be rejected.

**VII. THREATS TO VALIDITY**

The following sections describe the four types of threats [59], [63] that could potentially affect the validity of our results. These sections also describe how these threats were mitigated.

**A. INTERNAL VALIDITY**

The first factor that can affect the internal validity is related to the creativity, intellect, and competence of the student subjects. Some students could be more competent and efficient than other fellow students. In the same way, interest in programming languages, background knowledge, and relevant experience can also influence the overall performance of the students. Moreover, the performance of the student teams could also be affected by gender differences.



TABLE 11. MANOVA results for software quality.

Sr#	Phase	Independent Variables	Dependent Variables	Value	F	hyp df	err df	sig. (p-value)
1	Analysis and Design Phase	O-THI	Software Quality (Aggregated results of Und, Cor, Lay, and Rel)	10.48	8.15	9.00	7.00	<b>0.01</b>
2		C-THI		59.89	19.96	12.00	4.00	<b>0.01</b>
3		E-THI		3.17	1.90	10.00	6.00	0.22
4		A-THI		3360.66	480.09	14.00	2.00	<b>0.00</b>
5		N-THI		13.80	8.28	10.00	6.00	<b>0.01</b>
6	Implementation Phase	O-THI	Software Quality (Aggregated results of WSB, CC, DD, and MI)	8.67	6.19	14.00	10.00	<b>0.00</b>
7		C-THI		5.21	7.29	10.00	14.00	<b>0.00</b>
8		E-THI		57.50	41.07	14.00	10.00	<b>0.00</b>
9		A-THI		5.53	2.76	16.00	8.00	0.07
10		N-THI		10.22	8.65	13.00	11.00	<b>0.00</b>
11	Testing Phase	O-THI	Software Quality (Aggregated results of DU, RC, and TCC)	4.38	2.19	14.00	7.00	0.15
12		C-THI		6.69	3.34	14.00	7.00	0.06
13		E-THI		3.03	4.93	8.00	13.00	<b>0.01</b>
14		A-THI		4.31	1.73	15.00	6.00	0.26
15		N-THI		5.29	3.96	12.00	9.00	<b>0.02</b>

df=degrees of freedom, sig.= Significance of F, hyp=hypothesis, err=Error, Und=Understandability, Cor=Correctness, Lay=Layout, Rel=Appropriately Used Relationships, WSB=Weighted Sum of Bugs, CC=Cyclomatic Complexity, DD=Defect Density, MI=Maintainability Index, DU=Defects Uncovered, RC=Requirements Coverage, TCC=Test Case Conformity (p-value>=0.05)

TABLE 12. ANOVA results for team productivity.

Sr#	Phase	Independent Variables	Dependent Variable	df	SS	MS	F	sig. (p-value)
1	Analysis and Design Phase	O-THI	Productivity	1.00	14.29	14.29	2.28	0.15
2		C-THI		1.00	18.04	18.04	3.00	0.10
3		E-THI		1.00	1.86	1.86	0.26	0.62
4		A-THI		1.00	40.90	40.90	9.10	<b>0.01</b>
5		N-THI		1.00	0.00	0.00	0.00	0.99
6	Implementation Phase	O-THI	Productivity	1.00	0.12	0.12	2.11	0.16
7		C-THI		1.00	0.19	0.19	3.60	0.07
8		E-THI		1.00	0.48	0.48	12.03	<b>0.00</b>
9		A-THI		1.00	0.01	0.01	0.14	0.71
10		N-THI		1.00	0.10	0.10	1.68	0.21
11	Testing Phase	O-THI	Productivity	1.00	0.00	0.00	0.00	0.95
12		C-THI		1.00	1.06	1.06	4.82	<b>0.04</b>
13		E-THI		1.00	0.01	0.01	0.03	0.86
14		A-THI		1.00	0.72	0.72	3.04	0.10
15		N-THI		1.00	0.99	0.99	4.44	<b>0.05</b>

df=degrees of freedom, SS=Sum of Squares, MS=Mean Squares, sig.= Significance of F (p-value>=0.05)

To mitigate the above-mentioned threat, we formed students' teams by dividing students into three CGPA groups (Group A: 3.00 – 4.00, Group B: 2.50 – 2.99, and Group C: 2.00 – 2.49). Teams were formed by selecting one member from “Group A”, two members from “Group B”, and

two members from “Group C”. Moreover, we selected these members randomly from different groups. Besides this, we made sure that no more than one member in each team had some previous experience or domain knowledge. It was also decided to include no more than one female member in each

team. Last, but not the least, we also ensured that the average CGPA of each team lied between 2.50-2.80.

In the case of professionals, factors like qualifications, work experience, gender, and age can influence the results of this research. To minimize the impact of difference in qualifications, it was made sure that each team member had at least a bachelor's degree in Computer Science. It was also decided to select team members in the age range of 21 to 31 years to minimize the impact due to age differences. Furthermore, it was made sure that every team member has work experience in the range of 2 to 6 years. Last, but not the least, the impact of gender differences was minimized by ensuring that no more than one female member is present in a single team.

### B. EXTERNAL VALIDITY

To improve the generalizability of our results, we conducted our experiments in both academic as well as industrial environments. Furthermore, these experiments were conducted for three different phases of SDLC i.e. analysis and design, implementation, and testing.

### C. CONSTRUCT VALIDITY

We have used an IPIP 50-item personality test for personality traits identification. In this test, the subjects need to select a value for each of the five different factors of personality. A major problem with this kind of test is that participants can give fake responses to hide their true personalities. We mitigated this threat by assuring the participants that their personality profile will be treated as confidential and only be used for research purposes.

The use of the same project for both types of subjects (i.e. students and professionals) could affect the quality of our results. Therefore, we selected a simple project (i.e. MyShop) for students and a relatively large and complex project (i.e. Online Job Portal) for professionals.

Another issue relates to the constructs used to represent the dependent variable (i.e. software quality and team productivity). In our experiments, students' teams' performance in their term projects were used as team productivity. The measures of this may also be affected by third party variables such as learning strategies, cognitive ability, or self-motivation. One might presume that there was significant home studying between lab sessions. We believe that this did not really occur and students relied mostly on what they were presented in the tutorials and in the classes. Furthermore, software quality (the other dependent variable) may be compromised as students are in their learning phase. We tried to mitigate the impact of this factor by choosing a relatively simple project for students' teams (i.e. MyShop).

### D. CONCLUSION VALIDITY

The chances of Type 2 error increase when we have low statistical power. Tabachnick and Fidell [64] claim that a sample size of at least 20 in a group should ensure "robustness". During these experiments, we used a large pool of subjects

(with 215 students and 35 professionals) thus reducing the likelihood of committing a Type 2 error and conducted ANOVA and MANOVA tests to ascertain the statistical significance of our results. The use of the dichotomization process on our independent variables may increase the chance of false positive results i.e. type I error. To avoid this threat, we have only considered the values with 95% or above confidence level i.e.  $p\text{-value} \geq 0.05$ .

## VIII. IMPLICATIONS OF RESEARCH

Different implications can be drawn from the outcomes of this research. The following sections describe these implications for three different stakeholders i.e. practitioners, educators, and researchers.

### A. IMPLICATION FOR PRACTITIONERS

Our research has quantitatively studied and analyzed the impact of trait-wise team homogeneity on team productivity and software quality. Our results reveal that, in the case of professionals, teams with higher values of O-THI, C-THI, E-THI, and A-THI perform better in all three phases of the SDLC. They were not only more productive but were also found to produce better quality analysis and design models, write better quality code, and document better quality test cases. This implies that considering these personality traits while hiring new employees for specific projects and while assigning tasks to existing employees would be beneficial for software development managers.

### B. IMPLICATION FOR EDUCATORS

Our results show that the students' teams with greater values of O-THI, C-THI, and A-THI performed better during the analysis and design phase. Students' teams with higher values of O-THI, C-THI and E-THI performed better during the implementation phase while those with higher values of C-THI and A-THI performed better in the testing phase. This implies that these metrics can be used as a pedagogical tool for different courses in the software engineering stream. Faculty members can consider personality traits and their corresponding trait-wise THIs while forming student teams for group-based assignments, term projects, and even capstone (i.e. senior year) projects.

### C. IMPLICATION FOR RESEARCHERS

This research focused on the broad-level personality traits of the Five-Factor Model (FFM). We did not take lower-level/detailed personality traits (called facets) into consideration. Researchers can do personality profiling using these facets as well to gain more knowledge about personality attributes and their impact on software development. So far, we have also not assessed the impact of trait-wise team homogeneity on factors like conflicts resolution and satisfaction level. Researchers may, therefore, evaluate the impact of trait-wise team homogeneity on these factors. Furthermore, researchers can also study the impact of team homogeneity on other phases of SDLC such as requirements engineering and maintenance.

## IX. CONCLUSION AND FUTURE WORK

The main goal of this research was the quantification of the concept of trait-wise team homogeneity. For this purpose, five different metrics i.e. O-THI, C-THI, E-THI, A-THI, and N-THI were introduced. The impact of these five metrics on software quality and team productivity was evaluated quantitatively by conducting experiments in academic as well as industrial environments during three different phases of SDLC i.e. analysis and design, implementation, and testing. Our results reveal that software quality and team productivity are influenced by different combinations of these five trait-wise team homogeneity metrics for different phases of SDLC and for different types of subjects (students or professionals). These results have utility for both practitioners as well as educationists who are involved in team formation and task allocation.

This research is just the first step to quantify the concept of trait-wise team homogeneity and there is a lot of room for future research. For instance, so far, we have used only the IPIP inventory in our research for personality profiling. It would also be worthwhile to use other variants of IPIP i.e. 120-item IPIP NEO-PI-R [47], NEO-PI-R Domains [48], NEO-PI-R Facets [48], etc. Similarly, other personality models/tools (apart from FFM) can also be utilized e.g. MBTI, KTS, 16PF, ACL, etc. Last, but not the least, these experiments may be replicated using teams of different sizes and projects of different scale, domains, and complexity to check the generalizability of our results.

## REFERENCES

- [1] D. Varona, L. F. Capretz, Y. Piñero, and A. Raza, "Evolution of software engineers' personality profile," *ACM SIGSOFT Softw. Eng. Notes*, vol. 37, no. 1, pp. 1–5, Jan. 2012.
- [2] R. S. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach*. New York, NY, USA: McGraw-Hill, 2014, p. 651.
- [3] J. A. Lima and G. Elias, "Selection and allocation of people based on technical and personality profiles for software development projects," in *Proc. 15th Latin Amer. Comput. Conf. (CLEI)*, Sep. 2019, pp. 1–10.
- [4] T. DeMarco and T. Lister, *Peopleware: Productive Projects and Teams*. New York, NY, USA: Dorset House, 1999.
- [5] B. W. Boehm, *Software Cost Estimation With COCOMO II*. Upper Saddle River, NJ, USA: Prentice-Hall, 2000.
- [6] P. Lenberg, R. Feldt, and L. G. Wallgren, "Behavioral software engineering: A definition and systematic literature review," *J. Syst. Softw.*, vol. 107, pp. 15–37, Sep. 2015.
- [7] E. Weilemann, "A winning team—What personality has to do with software engineering," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng., Companion Proc. (ICSE-Companion)*, May 2019, pp. 252–253.
- [8] Z. Akarsu and M. Yilmaz, "Managing the social aspects of software development ecosystems: An industrial case study on personality," *J. Softw., Evol. Process*, vol. 32, no. 11, p. e2277, Nov. 2020.
- [9] F. F. Mendes, E. Mendes, and N. Salleh, "The relationship between personality and decision-making: A systematic literature review," *Inf. Softw. Technol.*, vol. 111, pp. 50–71, Jul. 2019.
- [10] V. C. Kuko, D. Music, Z. Vejzovic, and J. Azemovic, "Model of foresight work habits of agile software team members by personality traits," in *Proc. Int. Conf. Comput., Inf. Telecommun. Syst. (CITS)*, Aug. 2019, pp. 1–5.
- [11] S. T. Acuña, M. N. Gómez, J. E. Hannay, N. Juristo, and D. Pfahl, "Are team personality and climate related to satisfaction and software quality? Aggregating results from a twice replicated experiment," *Inf. Softw. Technol.*, vol. 57, no. 1, pp. 141–156, Jan. 2015.
- [12] N. Qamar and A. A. Malik, "Birds of a feather gel together: Impact of team homogeneity on software quality and team productivity," *IEEE Access*, vol. 7, pp. 96827–96840, 2019.
- [13] M. O. Yilmaz, R. V. O'Connor, R. Colomo-Palacios, and P. Clarke, "An examination of personality traits and how they impact on software development teams," *Inf. Softw. Technol.*, vol. 86, no. 1, pp. 101–122, Jun. 2017.
- [14] L. A. Pervin, *Personality: Theory, Assessment and Research*. Hoboken, NJ, USA: Wiley, 1989.
- [15] I. B. Myers, L. K. Kirby, and K. D. Myers, "An introduction to types: A guide to understanding your results on the Myers-Briggs types indicator," CPP, Palo Alto, CA, USA, Tech. Rep., 1998.
- [16] R. R. McCrae and O. P. John, "An introduction to the five-factor model and its applications," *J. Pers.*, vol. 60, no. 2, pp. 175–215, 1992.
- [17] D. Keirse and M. M. Bates, "Please understand me," Prometheus Nemesis, Tech. Rep., 1984.
- [18] S. Cruz, F. Q. B. Silva, and L. F. Capretz, "Forty years of research on personality in software engineering: A mapping study," *Comput. Hum. Behav.*, vol. 46, no. 1, pp. 94–113, May 2015.
- [19] J. M. Levine and R. L. Moreland, "Progress in small group research," *Annu. Rev. Psychol.*, vol. 41, pp. 585–634, Dec. 1990.
- [20] S. L. Kichuk and W. H. Wiesner, "The big five personality factors and team performance: Implications for selecting successful product design teams," *J. Eng. Technol. Manage.*, vol. 14, nos. 3–4, pp. 195–221, Sep. 1997.
- [21] M. A. G. Peeters, H. F. J. M. V. Tuijil, C. G. Rutte, and I. M. M. J. Reymen, "Personality and team performance: A meta analysis," *Europ. J. Pers.*, vol. 20, pp. 377–396, Aug. 2006.
- [22] J. Karn and T. Cowling, "A follow up study of the effect of personality on the performance of software engineering teams," in *Proc. ACM/IEEE Int. Symp. Emp. Soft. Eng.*, Jun. 2006, pp. 232–241.
- [23] V. Pieterse, D. G. Kourie, and I. P. Sonnekou, "Software engineering team diversity and performance," in *Proc. South Afr. Inst. Comp. Scientists Inf. Technol. (SAICSIT)*, 2006, pp. 180–186.
- [24] A. S. Barroso, J. S. Madureira, M. S. Soares, and R. P. C. do Nascimento, "Influence of human personality in software engineering—A systematic literature review," in *Proc. 19th Int. Conf. Enterprise Inf. Syst.*, 2017, pp. 53–62.
- [25] J. Gulati, P. Bhardwaj, B. Suri, and A. S. Lather, "A study of relationship between performance, temperament and personality of a software programmer," *ACM SIGSOFT Softw. Eng. Notes*, vol. 41, no. 1, pp. 1–5, Feb. 2016.
- [26] P. Sfetsos, I. Stamelos, L. Angelis, and I. Deligiannis, "An experimental investigation of personality types impact on pair effectiveness in pair programming," *Empirical Soft. Eng.*, vol. 14, no. 1, pp. 187–226, Apr. 2009.
- [27] J. Karn, S. Syed-Abdullah, A. J. Cowling, and M. Holcombe, "A study into the effects of personality type and methodology on cohesion in software engineering teams," *Behav. Inf. Technol.*, vol. 26, no. 2, pp. 99–111, Mar. 2007.
- [28] K. S. Choi, F. P. Deek, and I. Im, "Exploring the underlying aspects of pair programming: The impact of personality," *Inf. Softw. Technol.*, vol. 50, no. 11, pp. 1114–1126, Oct. 2008.
- [29] A. R. Gilal, J. Jaafar, L. F. Capretz, M. Omar, S. Basri, and I. A. Aziz, "Finding an effective classification technique to develop a software team composition model," *J. Soft., Evol. Process*, vol. 29, no. 10, pp. 1–12, 2017.
- [30] R. Poonam and C. M. Yasser, "An experimental study to investigate personality traits on pair programming efficiency in extreme programming," in *Proc. 5th Int. Conf. Ind. Eng. Appl. (ICIEA)*, Apr. 2018, pp. 96–99.
- [31] Z. U. Kamangar, U. A. Kamangar, Q. Ali, I. Farah, S. Nizamani, and T. H. Ali, "To enhance effectiveness of crowdsourcing software testing by applying personality types," in *Proc. 8th Int. Conf. Softw. Inf. Eng.*, Apr. 2019, pp. 15–19.
- [32] A. S. Barroso, K. H. de J. Prado, M. S. Soares, and R. P. C. do Nascimento, "How personality traits influences quality of software developed by students," in *Proc. 15th Brazilian Symp. Inf. Syst.*, May 2019, pp. 1–8.
- [33] N. Qamar and A. A. Malik, "Impact of design patterns on software complexity and size," *Mehran Uni. Res. J. Eng. Technol.*, vol. 39, no. 2, pp. 342–352, Apr. 2020.
- [34] T. Walle and J. E. Hannay, "Personality and the nature of collaboration in pair programming," in *Proc. 3rd Int. Symp. Empirical Softw. Eng. Meas.*, Oct. 2009, pp. 203–213.
- [35] J. E. Hannay, E. Arisholm, H. Engvik, and D. I. K. Sjoberg, "Effects of personality on pair programming," *IEEE Trans. Softw. Eng.*, vol. 36, no. 1, pp. 61–80, Jan. 2010.
- [36] J. Chao and G. Atli, "Critical personality traits in successful pair programming," in *Proc. AGILE (AGILE)*, 2006, pp. 89–93.

- [37] R. Feldt, L. Angelis, R. Torkar, and M. Samuelsson, "Links between the personalities, views and attitudes of software engineers," *Inf. Softw. Technol.*, vol. 52, no. 6, pp. 611–624, Jun. 2010.
- [38] T. Kanij, R. Merkel, and J. Grundy, "An empirical study of the effects of personality on software testing," in *Proc. 26th Int. Conf. Softw. Eng. Educ. Training (CSEET)*, May 2013, pp. 239–248.
- [39] N. Salleh, E. Mendes, and J. Grundy, "Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments," *Empirical Softw. Eng.*, vol. 19, no. 3, pp. 714–752, Jun. 2014.
- [40] I. Lykourantzou, A. Antoniou, Y. Naudet, and S. P. Dow, "Personality matters: Balancing for personality types leads to better outcomes for crowd teams," in *Proc. 19th ACM Conf. Comput.-Supported Cooperat. Work Social Comput.*, Feb. 2016, pp. 260–273.
- [41] M. Shameem, C. Kumar, and B. Chandra, "A proposed framework for effective software team performance: A mapping study between the team members' personality and team climate," in *Proc. Int. Conf. Comput., Commun. Autom. (ICCCA)*, May 2017, pp. 912–917.
- [42] A. S. Barroso, J. S. Madureira, T. D. S. Souza, B. S. D. A. Cezario, M. S. Soares, and R. P. C. do Nascimento, "Relationship between personality traits and software quality—big five model vs. object-oriented software metrics," in *Proc. 19th Int. Conf. Enterprise Inf. Syst.*, 2017, pp. 63–74.
- [43] A. Amin, S. Basri, M. Rehman, L. F. Capretz, R. Akbar, A. R. Gilal, and M. F. Shabbir, "The impact of personality traits and knowledge collection behavior on programmer creativity," *Inf. Softw. Technol.*, vol. 128, Dec. 2020, Art. no. 106405.
- [44] R. Yan, W. R. Ringwald, J. Vega, M. Kehl, S. W. Bae, A. K. Dey, C. A. Low, A. G. C. Wright, and A. Doryab, "Exploratory machine learning modeling of adaptive and maladaptive personality traits from passively sensed behavior," *Future Gener. Comput. Syst.*, vol. 132, pp. 266–281, Jul. 2022.
- [45] M. A. Kosan, H. Karacan, and B. A. Urgen, "Predicting personality traits with semantic structures and LSTM-based neural networks," *Alexandria Eng. J.*, vol. 61, no. 10, pp. 8007–8025, Oct. 2022.
- [46] P. T. Costa and R. R. McCrae, "NEO five-factor inventory (NEO-FFI)," *Psycho. Assess. Resour.*, Tech. Rep., 1989.
- [47] P. T. Costa and R. R. McCrae, "NEO PI-R professional manual," *Psycho. Assess. Resour.*, Tech. Rep., 1992.
- [48] L. R. Goldberg, "The international personality item pool and the future of public-domain personality measures," *J. Res. Personality*, vol. 40, no. 1, pp. 84–96, Feb. 2006.
- [49] S. Boslaugh, *Statistics in a Nutshell*. Sebastopol, CA, USA: O'Reilly Media, 2012.
- [50] V. R. Basili, F. Shull, and F. Lanubile, "Building knowledge through families of experiments," *IEEE Trans. Softw. Eng.*, vol. 25, no. 4, pp. 456–473, Jul. 1999.
- [51] V. R. Basili and H. D. Rombach, "The TAME project: Towards improvement-oriented software environments," *IEEE Trans. Softw. Eng.*, vol. SE-14, no. 6, pp. 758–773, Jun. 1988.
- [52] S. L. Pfleeger, "Experimental design and analysis in software engineering," *Ann. Softw. Eng.*, vol. 1, no. 1, pp. 219–253, 1995.
- [53] T. McCabe, "A complexity measure," *IEEE Trans. Softw. Eng.*, vol. SE-2, no. 4, pp. 308–320, Dec. 1976.
- [54] K. D. Welker, "The software maintainability index revisited," *CrossTalk, J. Defense Soft. Eng.*, vol. 14, pp. 18–21, Aug. 2001.
- [55] *PhpMetrics*. Accessed: Feb. 22, 2020. [Online]. Available: <https://phpmetrics.org/>
- [56] *Trait-Wise Team Homogeneity Index (THI) Calculator*. Accessed: Mar. 22, 2020. [Online]. Available: <http://zelogix.com/personality/admin/>
- [57] *TimeKeeper Tool*. Accessed: Mar. 22, 2020. [Online]. Available: <http://zelogix.com/personality/timelog/>
- [58] N. Qamar and A. A. Malik, "Evaluating the impact of pair testing on team productivity and test case quality—A controlled experiment," *Pak. J. Eng. Appl. Sci.*, vol. 25, pp. 80–88, Dec. 2019.
- [59] C. Wohlin, P. Runeson, M. Höst, M. Ohlson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering: An Introduction*. Norwell, MA, USA: Kluwer, 2000.
- [60] *IBM SPSS Statistics*. Accessed: Mar. 22, 2020. [Online]. Available: <https://www.ibm.com/products/spss-statistics>
- [61] P. N. Tang, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Reading, MA, USA: Addison-Wesley, 2006.
- [62] N. Qamar and A. A. Malik, "Determining the relative importance of personality traits in influencing software quality and team productivity," *Comput. Inform.*, vol. 39, no. 5, pp. 1001–1028, 2020.
- [63] W. R. Shadish, T. D. Cook, and D. T. Campbell, *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Boston, MA, USA: Houghton, Mifflin and Company, 2002.
- [64] B. G. Tabachnick and L. S. Fidell, *Using Multivariate Statistics*, 4th ed. Boston, MA, USA: Allyn and Bacon, 2002.



**NOSHEEN QAMAR** received the Ph.D. degree in computer science from the National University of Computer and Emerging Sciences (FAST-NUCES), Lahore, Pakistan, in 2020. She is currently working as an Assistant Professor at the Department of Software Engineering, University of Management and Technology, Lahore. Before joining the academia, in 2015, she has gained seven years of industry experience with last job title as a Software Development Manager. Her research interests include software engineering, design patterns, software teams, project management, empirical software engineering, and requirements engineering.



**ALI AFZAL MALIK** (Senior Member, IEEE) received the M.S. and Ph.D. degrees in computer science from the University of Southern California (USC), Los Angeles, CA, USA, in 2007 and 2010, respectively. He is currently working as an Assistant Professor of computer science with the National University of Computer and Emerging Sciences (FAST-NUCES). He started his professional career, in 2003, working as a Software Engineer with Techlogix—a well-reputed Pakistani software house. Before joining FAST-NUCES, in 2013, he has held an Adjunct Faculty position at the Lahore University of Management Sciences (LUMS) and a full-time faculty position at the University of Central Punjab (UCP). He has undertaken research in software cost estimation at two of the world's leading research centers in software engineering i.e. USC's Center for Systems and Software Engineering (CSSE) and Institute of Software, Chinese Academy of Sciences (ISCAS). His current research interests include empirical software engineering, requirements engineering, and software cost estimation. He received the prestigious Fulbright Scholarship, in 2005. He was awarded the Office of International Services (OIS) Academic Achievement Award twice (2007 and 2010) during his stay at USC. His research paper on the quantitative aspects of requirements elaboration was given the Best Paper Award in SBES 2008 (Sao Paulo, Brazil).

• • •