JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 33, 1481-1500 (2017)
DOI: 10.6688/JISE.2017.33.6.6

Risk Aware and Quality Enriched Effort Estimation for Mobile Applications in Distributed Agile Software Development

WAQAR ASLAM[1,*], FARAH IJAZ[1], M. IKRAMULLAH LALI[2] AND WAQAR MEHMOOD[3]
[1]*Department of Computer Science and Information Technology*
The Islamia University of Bahawalpur
Bahawalpur, 63100 Pakistan
[2]*Department of Software Engineering*
University of Gujrat
Gujrat, 50700 Pakistan
[3]*Department of Computer Science*
COMSATS Institute of Information Technology
Wah, 47040 Pakistan
E-mail: waqar.aslam@iub.edu.pk[*]; farah.ijaz@yahoo.com;
ikramlali@gmail.com; drwaqar@ciitwah.edu.pk

Mobile application development has been a promising field in Software Engineering (SE) for more than the last decade. It is growing rapidly due to an ever increasing popularity of smart phones. Due to benefits it offers, SE in general, is trending towards Agility based approaches and distributed development setups. These benefits include development cost reduction, predictability, visibility, changeability and availability of expertise in specialized domains due to offshore teams. Optimal deployment of resources has a key role in the success of a project. To this aim, accurate effort estimation is a vital activity. Effort estimation method should consider artifacts such as complexity of tasks and prior experience of team members in the project domain. Agility and distributed development are two dimensions that are currently changing the development philosophies and shaping the development team profiles. Due to lack of experience in these two dimensions and ever changing nature of mobile application domain, any effort estimation method suffers from uncertainty. Thus the accuracy of effort estimation methods is comprised. This work ranks dealing with this uncertainty as a major challenge. To address it, the uncertainty is translated to associated risk whose consideration contributes towards accuracy of effort estimation. Thus we improve on risk-agnostic effort estimation methods by classifying and including risks in effort estimation, especially those which arise due to agility and distributed development for mobile application domain. Classification helps to understand the risk types involved and enables to better estimate their impact. Other than risk, we also consider novelty level and type of a project. Another contribution of this work is inclusion of quality aspect in software development, without which effort estimation methods map to single effort values. This limitation does not allow trading-off between chosen quality levels and corresponding effort values. Enabling software development at multiple quality levels overcomes this limitation and offers liberty of objective optimizations such as maximal quality and minimal time needed for project completion.

Keywords: effort estimation, model, distributed agile software development, risk, quality

1. INTRODUCTION

Rich Mobile Applications (RMAs) domain is catching lots of research attention from both industry and academic sectors. Richness of RMAs refers to improved user

Received January 26, 2017; revised February 24, 2017; accepted August 10, 2017.
Communicated by Tzung-Pei Hong.

experience that suits well to the individual needs and habits, and to the societal trends and adaptations. RMAs are considered to be the proposal of future smart phones seeking to offer high functionalities. RMAs are still in early stages and require more attention [1]. In this pre-RMA era, complementary requirements are generally less addressed by traditional software methodologies. These include inter-operability between native and mobile web applications, ability to handle sensors, portability over a variety of software and hardware platforms, user interfaces of choice, complexity of testing, security and scarcity of resources such as power, computation and memory. The success of mobile applications, like other software applications, is increasingly dependent on non-functional requirements, also known as quality requirements [2]. RMA development has a lot of business appeal these days. Its success relies on proper decision making based on accurate effort estimation, which offers predictability on allocation and deployment of resources.

Presently various software development approaches are in practice, such as software development in a globally distributed manner known as Global Software Development (GSD), software development based on Agile Methods also formally known as Agile Software Development (ASD), and Distributed Agile Software Development (DASD) [3].

Given that Agile manifesto concentrates on collaboration and cooperation of all stake holders, triggering actions on changes, requires change management and traceability. The focus is on working software rather than on adopting processes. Due to adaptability to changes in a software project, ASD has become a great success, and it is now an accepted framework for software industries. In ASD, Software is developed incrementally in small iterations. On other hand, a practical GSD scenario requires involvement of distributed teams that are dispersed geographically with possible different time zones and cultures. Executing Agile manifesto in GSD gives rise to DASD, which offers challenges that span on both paradigms. In DASD projects are developed out of various parts, where each part may be developed at a separate site. Benefitting from such a setup is not straightforward and leads to compromises such as those arising due to technical limitations of teams operating at various locations and state laws [4].

There are various Agile Methods that follow the common manifesto of changeability preference over fixed planning. One such method is Scrum that exhibits management due to which it has a higher success rate, hence it is adopted widely. A recent survey (January 2016 to June 2016) of mobile application development companies conclude on the popularity of Scrum. It is noted that the basic principle used for mobile applications development is 'regular meetings of mobile applications development team', which reflects on Scrum methodology. Positive feedbacks offer better management and control, development speed, maintainability of project interactions, improved communication and performance, and more transparency [5]. Agile development is also tailored to be used for mobile application development, such as Mobile-D [6]. Another lightweight approach is proposed to estimate the method level energy consumption in support of Android applications [7].

Effort estimation is a project management activity that is essential to deal efficiently with resources [3]. An inaccurate estimation creates many problems, like late delivery, over budget and delivered project not satisfying user requirements [8]. Mobile applications being a new domain, their project effort estimation can be made using an empirical analogy based process. Another popular technique is Planning Poker [3, 8-10]. It is based on expert judgments employed extensively within Agile Methods. In Planning

Poker, Story Points are estimated collectively by experts, which are units to represent software features. Such collective opinions alleviate the chances of over-optimism and reduce the issues of anchoring and strong personalities [9]. A Story Point generally corresponds to an ideal day of work. Often predefined set of possible Story Points are used to estimate User Stories, *e.g.*, 0.5, 1, 2, 3, 5, 8, 13, 20, 40 and 100. A Chi-Square methodology is proposed for deviation control of project plan [11]. Identification and quantitative analysis of large scale project success factors is done in [12]. Empirical investigation on effort estimation in DASD points out an absence of a technique that considers distributed aspect of development [3]. In Scrum, User Stories are defined by the Product Owner, while the estimations are made by the Scrum development team [9].

The existing methods for effort estimation consider size, complexity and priority as independent variants [8, 13]. Risks are effectively handled in a model proposed in [14]. To the best of our knowledge none of the existing works have focused on risk for effort estimation in DASD. Risk classification and corresponding estimations are also absent from the reported literature. Due to increased risks involved in various activities of DASD, separation between estimated effort and actual effort increases. Thus estimating optimal resource allocation a priori becomes in-deterministic. To improve on this limitation, we deem risk as an important independent variant whose detailed investigations brings more alignment between estimated effort and actual effort taken. To this aim, we contribute with an improved method that considers the implications of risk towards effort estimation in DASD. Our second contribution in this paper is the consideration of quality aspect in the delivered software. Given that achieving high quality requires more effort, its consideration allows a tradeoff between quality and time. This tradeoff can be used effectively by the project managers to offer choices to the project clients. Every day new mobile applications are conceived that may differ on criticality and lifetime of information. Hence SE techniques for them are quite in transition phase of being matured. Thus two more dimensions, namely novelty and type stand strong candidates for consideration of any effort estimation model, as in our case. Effectively, our proposed method of effort estimation seeks mapping, denoted by $f(\cdot)$[1] for each User Story:

$$f: \mathbb{N} \times \mathbb{N} \times \mathbb{R} \times \mathbb{N} \times \mathbb{N} \times \mathbb{R} \rightarrow \mathbb{R},$$

$$\textit{effort} = f(\textit{size}, \textit{complexity}, \textit{quality}, \textit{novelty}, \textit{type}, \textit{risk}). \quad (1)$$

Effort of a Sprint is the sum of efforts for all User Stories within it, while estimated effort of a Project is the sum of all Sprint efforts that are required to complete the Project.

2. LITERATURE REVIEW

Effort estimation in ASD is relatively a new area of focus, while not much work is reported in GSD. A comprehensive overview of the state-of-the-art in the field of effort estimation in ASD is presented in [8]. It identifies that Planning Poker is used frequently for effort estimation, while Story Points is the mostly used size metrics according to 25 selected studies. Another similar survey on effort estimation techniques, conducted on 60 practitioners from 16 different countries reveals the proportions of three techniques used in ASD: Planning Poker (63%), analogy (47%) and expert judgment (38%) [10]. It also

[1] First line of Eq. (1) is the signature of the mapping $f(\cdot)$. $\mathbb{N}$ and $\mathbb{R}$ denote the sets of natural numbers and real numbers, respectively.

assesses that Planning Poker based on Story Points, is mostly employed solo or in combination with other techniques and yields more estimation accuracy. Mostly effort estimation techniques are based on size estimation of User Stories. In this context, commonly used size estimation methods use Source Lines of Code (SLOC), Function Point Analysis (FPA) and Use Case Points [15].

Three qualitative surveys report results on effort estimation in ASD [10]. First one presents that projects using flexible process models (incremental, Agile) experience less overrun effort as compared to those using sequential models. Second one presents the advantage of customer collaboration in the form of lesser effort overruns. Third one highlights estimation related problems when developers deal with coarse grained User Stories.

Optimization techniques such as Genetic Algorithms, Particle Swarm, Artificial Neural Network and Fuzzy Inference Systems are applied successfully for effort and cost estimation in ASD. The applicability of these optimizations is for small projects, while further exploration is needed for medium and large projects [7].

Based on our review of existing works in effort estimation in ASD, we systematically identify relevantly recent models, processes, algorithms, metrics and adjusted Story Points based on various factors that affect the results. Mostly Business Value of the product is not considered and accuracy of method not found. The listing is presented in Table 1.

Table 1. A systematic identification of effort estimation techniques in ASD.

Year	Proposed Methodology	Effort Estimation	Velo-city	Business Value	Accuracy Metric
2010 [16]	Systematic estimation and dynamic tracking	✓	✓	×	–
Function and Story Points used; Kalman filter used for tracking project progress.					
2012 [17]	Pattern language	✓	✓	×	–
Pattern language for Agile estimation; eight patterns and their relationships are presented.					
2012 [9]	Empirical Study	✓	✓	×	BRE, BREbias
Accuracy of Planning Poker is compared with Statistical combination of individual estimations.					
2012 [13]	Effort estimation model	✓	✓	×	MMRE, PRED
Effort estimation model; tested on empirical data collected from 21 software projects; good results.					
2013 [18]	Expert judgment with regression analysis	✓	×	×	–
Effort estimation model (AgileMOW) for web based projects development using Agile method.					
2013 [19]	Set of metrics	✓	×	×	–
Proposed metrics for accuracy of effort estimation and tool for Planning Poker; inaccuracy in estimation identified.					
2013 [20]	Effort estimation model	✓	×	×	–
Enhancement of Use Case Point estimation model; focus on efficiency and risk factors for estimation in ASD.					
2014 [21]	Algorithm	✓	×	×	–
Algorithm considers various types of project and people related factors; focus on release date, effort, cost and time.					
2014 [22]	Examined PSO model	✓	✓	×	MRE, MMRE, (T/M)MMRE
Swarm Optimization-Based Framework for effort estimation in ASD.					
2014 [23]	Machine Learning with neural network	✓	×	×	MER
Proposed model considers complexity and environmental factors for CMMI organizations; estimation accuracy improved.					
2014 [24]	Algorithm	✓	✓	×	–
A mathematical technique for effort, time and cost estimations.					
2015 [8]	Adjusted Story Points	✓	✓	✓	–
Proposed method uses priority, size and complexity factors of User Stories.					
2015 [25]	Neural network based approach	✓	✓	×	MMRE, PRED, MSE,R2
Various neural networks (GRNN, PNN, GMDH, cascade) used; results empirically validated; cascade outperforms.					
2016 [26]	Machine Learning based approach	✓	×	×	MMRE
Method used with Machine Learning algorithms; an improvement on manual Planning Poker results.					

These days there is a trend of moving towards GSD to maintain costs. Some aspects of ASD become more challenging in DASD, hence the required effort increases [27]. To meet such challenges, the reported recommendation is to use Planning Poker as the basis of any technique developed. Effort estimation in DASD is also made using empirical research as reported in [3]. Participants belonging to 12 countries establish the popularity of Planning Poker as the basis for effort estimation. According to [27], when various effort estimation methods for GSD were applied for co-located projects, the real results were not encouraging, hence a need for improvement in this area. Similarly effort estimation techniques pointed out in [3], need to be improved for cases of GSD.

Conclusively the previous work does not focus on effort estimation in DASD. This work focuses in this area and contributes by considering various inherent risks that arise when project development shifts from co-located teams to distributed teams. Risk based estimation captures the real situation better, thus improves accuracy.

3. PROPOSED MODEL

DASD approach is followed to facilitate better software development solutions in terms of minimum cost and time. Different characteristics of distributed software development, when combined with principles and practices of Agile methods create unique adversities, offers advantages of both worlds but also gives rise to risk challenges [28]. Mainly these challenges are due to the fact that both worlds are mostly orthogonal and also in contrast to each other. Distributed software development requires communication between distributed teams formally, while Agile methods are based on informal interaction between team members. Agile methods focus on deliverability of functional software and adopt practices for such aims as, face to face interaction, retrospectives, self organizing teams, *etc.* When these practices are required to be followed in a distributed setup of teams, many challenges and risks arise. To this end, software development in DASD is risky as well as more complex to manage [4]. Based on our review of existing works (also including [29, 30]), it is concluded that:

'A software risk can be defined as an uncertainty of any condition or event that has negative impact on project in the form of software product with low quality, enhanced costs and delayed completion. In short, chance of project failure increases if software risk is ignored.'

In contrast with other engineering projects, SE involves risks inherently. Without considering various risks involved, disparity between estimated/actual effort (or cost or time) increases. A detailed analysis of risk is vital that is based on identifying its types, and impact/chances of each.

To overcome the deficiency due to ignoring risk, we propose an effort estimation model for mobile application development in DASD. We use basic factors such as Size and Complexity to estimate effort in User Stories [13] and advance factors such as Risk, Quality, Novelty and Type to better reflect on requirements due to DASD setups, especially applicable for mobile applications domain.

3.1 Size Factor and Complexity Factor

User Story size has foremost importance: it is given relatively in terms of User Story Points. Size is the basis of estimating any kind of User Story. Complexity factor is the second most important factor that has a high impact on effort required. Complexity has a tendency to introduce uncertainty to estimation. It can be defined in terms of number of dependencies on other User Stories and mutual effects. It depends on required skills that are significant but deficient in team. It is not trivial to describe properly complexity levels of User Stories. Table 2 lists values assigned to Size and Complexity of User Stories according to their nature [13].

3.2 Quality Factor

These include those attributes that are associated to software product quality. Mostly software quality requirements are specified as software measurable properties based on quality model given in the ISO/IEC 9126-1 standard [31], which is the most significant standard for quality assurance. Similarly for mobile application development, main recommended quality attributes include Functionality, Reliability, Usability, Efficiency, Maintainability and Portability [32]. For our case of mobile applications in DASD, we select quality attributes as: Reliability (qR), Usability (qU), Maintainability (qM), Portability (qP), Correctness (qC) and Performance (qF). Our choices aim at addressing compromises due to local optimizations in limited resources scenario of mobile devices (battery power, memory and processor speed). Highly prioritized User Stories with non-functional quality requirement should be developed accordingly. Highly prioritized User Stories tend to consume more resources. Thus remaining resources may be managed by allocation to low priority User Stories, to achieve a statistical balance of resources usage over the runtime of various scenarios.

Let $\overrightarrow{QW}$ be the vector of these attributes, then

$$\overrightarrow{QV} = [qR, qU, qM, qP, qC, qF]. \tag{2}$$

In this case, Quality is expressed by six attributes. Their relative priority can be expressed by assigning appropriate weights to them. Let $\overrightarrow{QW}$ be the corresponding weight vector,

$$\overrightarrow{QW} = [qw_1, ..., qw_6], \sum_{i=1}^{6} qw_i = 1, \tag{3}$$

then any combination of six attribute values can be conveniently expressed on a single dimension by taking dot product,

$$Quality = \overrightarrow{QV} \cdot \overrightarrow{QW} = \sum_{i=1}^{6} qv_i \cdot qw_i, \tag{4}$$

where qv_i is the ith dimension of $\overrightarrow{QV}$. Next we define four attributes according to [32] and introduce two more to meet our specific goal for mobile application development. Sub-factors of each attribute are also given.

Reliability: Reliability is a set of characteristics that allow ability of software to maintain its performance level according to stated requirements. Maturity, Fault tolerance and Recoverability are sub-factors of Reliability.

Usability: Usability is a set of characteristics that allows ease of use, individual appraisal of use stated by users. Understandability, Learnability and Operability are sub-factors of Usability.

Maintainability: Maintainability is a set of characteristics that allow specified modifications. Analyzability, Changeability, Stability and Testability are sub-factors of Maintainability.

Portability: Portability is a set of characteristics that allow software to be transferred from one to another environment. Adaptability, Installability, Conformance and Replaceability are sub-factors of Maintainability.

Correctness: It is the extent of adherence to specifications.

Performance: Given time and resources, performance in general implies ability on various dimensions such as latency and throughput.

Effort of a project increases with required Quality. Any quality requirement decision is heavily influenced by the Business Values of User Stories, which refers to the amount of returns. According to Scrum development, User Stories with high Business Value should be completed first. User Stories with high Business Value have high impact on reputation and have significance for most of customers [8]. To assess quality required for each User Story in the product backlog, we propose assigning a value to each attribute (Reliability, Usability, Maintainability, Portability, Correctness and Performance) from one to five according to the criterion it meets best. Table 3 lists criteria and corresponding values. Once all attributes have values assigned, the quality is calculated according to Eq. (4).

Table 2. User Story Size factor and Complexity factor: criteria and values.

Criteria (Size)	Criteria (Complexity)	Value
Extremely large	Extremely complex	5
Moderately large	Moderately complex	4
Normal	Normal complex	3
Small	Easily understood technicality	2
Very small	Very easy	1

Table 3. User Story Quality attributes: criteria and values.

Criteria (Quality)	Value
Extremely high value	5
High value	4
Moderate value	3
Small value	2
Very small value	1

3.3 Novelty Factor

Mobile applications are rapidly emerging these days. Internet of Things and Cyber Physical Systems aim for ubiquitousness, thus new trends are evolving that are not anticipated before. With such dynamicity, software development is affected as well. Latest

approaches such as DASD are not settled in general, and the situation becomes more uncertain for the case of mobile applications. To accommodate this uncertainty in its own merit, Novelty factor of each User Story is also considered. According to the nature of User Story, its Novelty factor is chosen from three levels: one, extremely innovative (having no development experience and more research required), two, moderately innovative (having no working experience but expected it will be easy), three, simple innovative (having little bit prior experience related to this work). For assigning Novelty factor value, we propose scanning list for best match, starting from one to three. Novelty factor criteria and values are listed in Table 4.

Table 4. User Story Novelty and Type factors: criteria and values.

Criteria (Novelty)	Criteria (Type)	Value
Extremely innovative Story	Extremely Sensitive Story	3
Moderately innovative Story	Moderately Sensitive Story	2
Simple innovative Story	Simple Sensitive Story	1

3.4 Type Factor

Type of work also effect required effort for development. Here type refers to sensitivity of User Story. A sensitive User Story requires extremely high accuracy level; hence more effort for development than other type of software features. Type factor of User Story is assigned values from three levels: one, extremely sensitive User Story (required extremely high level of accuracy), two, moderately sensitive User Story (required high level of accuracy) and simple sensitive User Story. Again, Type factor be chosen, starting from one to three. Type factor criteria and values are listed in Table 4.

3.5 Risk Model

In this section, risk factors for DASD are identified and assigned values. Only those risk factors are included that explicitly or implicitly affect the effort estimation activity for DASD. Various risks in Distributed Software Development and DASD are pointed out in [4, 28]. We consider risks from these studies and categorize them into seven types: Sprint related risks, Team Management Related Risks, Team Interaction Related Risks, Communication Related Risks, Location Related Risks, Geographical Distribution Related Risks and Task Related Risks. For each type, its sub-types are also identified. The list is given in Table 5.

Sprint Related Risks (R1): If Sprint related risks (unclear objective of Sprint, lack of common understanding of User Stories and lack of visibility of project) are occurring, they will have an explicit effect on development effort.

Team Management Related Risks (R2): These risks (interdependencies between teams, lack of awareness between teams and lack of coordination between multiple teams) can affect development effort if teams are dependent on each other due to their work dependencies.

Table 5. Risk classification and sub-classification: risk type *t* is denoted as R*t* (bold) in the table, while risk type *t* and sub-type *s* is denoted as R*t.s*.

Sprint Related Risks (R1)	Team Management Related Risks (R2)	Team Interaction Related Risks (R3)	Communication Related Risks (R4)	Location Related Risks (R5)	Geographical Distribution Related Risks (R6)	Task Related Risks (R7)
Unclear objectives of Sprint (R1.1)	Dependencies between teams (R2.1)	Lack of face to face communication between teams (R3.1)	Different cultures (R4.1)	Changes in locations (R5.1)	Spatial distribution (R6.1)	Task uncertainty (R7.1)
Lack of common understanding of User Story (R1.2)	Lack of awareness between teams (R2.2)	Lack of trust between teams (R3.2)	Different languages (R4.2)	Changes in number of team locations (R5.2)	Temporal distribution (R6.2)	Task dependencies between distributed teams (R7.2)
Lack of visibility of project progress (R1.3)	Lack of coordination between teams (R2.3)	Lack of tool support (R3.3)	Poor communication skills (R4.3)	Location migration of team members (R5.3)	Goal distribution (R6.3)	Poor task knowledge among distributed teams (R7.3)

Team Interaction Related Risks (R3) and **Communication Related Risks (R4):** These risks (lack of face to face communication between teams, lack of trust between teams and lack of tool support) and communication related risks (different culture, different language and poor communication skills) affect negatively requirement gathering and prioritization, which become causes of unclear objectives of project, Sprint and User Story. Hence normal development effort increases.

Location Related Risks (R5): These risks (change in locations, numbers of locations and migration of team members from one location to another) also affect the development effort if they occur when teams start coordinated development for specific Sprints.

Geographical Distribution Related Risks (R6): These risks (spatial distribution, temporal distribution and goal distribution) gives rise to poor coordination between multiple teams at different locations and affect development effort negatively.

Task Related Risks (R7): These risks (task uncertainty, task dependencies between teams sited at different locations and poor knowledge about tasks among distributed teams) trigger lack of coordination between teams, hence explicitly affect development effort.

We consider all of these risks and their sub-types for our analysis with an aim to incorporate the specificity due to mobile application domain and DASD setup. Per risk theory, impact of each risk and its chances of occurrence together determine a risk value. For this matter, Table 6 gives a list of impact and corresponding probability. Given a risk sub-type, its impact and probability pair is selected.

Next we introduce some notations that are necessary for quantification of risk:

Table 6. Risk impact and probability of occurrence levels.

Impact Description	Probability	Level
No impact	(0,0.1]	1
Minimal impact	(0.1,0.2]	2
Normal impact but capable to complete within available resources	(0.2,0.3]	3
Extra resources required but capable to meet due dates	(0.3,0.4]	4
Small deviation in key milestones but capable to meet up due dates	(0.4,0.5]	5
Small deviation in key milestones; may not be able to meet due dates	(0.5,0.6]	6
Normal deviation in key milestones; not able to meet due dates	(0.6,0.7]	7
Major deviation in key milestones; Sprint targets affected	(0.7,0.8]	8
Some Sprint targets cannot be achieved	(0.8,0.9]	9
Extreme deviation; cannot achieve all Sprint targets	(0.9,1.0]	10

- Risk factor of User Story u: $Rf(u)$;
- Risk type: t;
- Risk sub-type: s;
- Impact of risk type t, sub-type s for User Story u: $iRt.s(u)$;
- Probability of risk type t, sub-type s for User Story u: $pRt.s(u)$;
- Maximum impact of risk sub-types for User Story u: $\max\_iRt.s(u)$;
- Maximum probability of risk sub-types for User Story u: $\max\_pRt.s(u)$;
- Risk normalization factor for User Story u: $RN(u)$.
- The estimated Story Points for User Story u without inclusion of risk factor: $esp(u)$;
- The estimated Story Points for User Story u with inclusion of risk factor: $ESP(u)$.

We assume that risk factor is proportional to the effort required otherwise. This aspect is captured as

$$ESP(u) = esp(u) + Rf(u). \tag{5}$$

Now $Rf(u)$ is stated as a product of weighted Story Point estimation without risk and normalized sum of all Risk values for User Story u:

$$Rf(u) = esp(u) \times w(u) \times \frac{1}{RN(u)} \sum_{t=1}^{7} \sum_{s=1}^{3} iRt.s(u) \times pRt.s(u), \tag{6}$$

where $0 < w(u) < 1$ is the weight of estimated effort without risk consideration and $RN(u)$ is the risk normalization factor used to keep risk value within [0, 1].

$$RN(u) = \sum_{t=1}^{7} \sum_{s=1}^{3} \max\_iRt.s(u) \times \max\_pRt.s(u). \tag{7}$$

$w(u)$ gives independence of risk weight per User Story.

3.6 Estimated Effort

Introducing few more notations:

- Size factor of User Story u: $Sf(u)$;
- Complexity factor of User Story u: $Cf(u)$;
- Quality factor of User Story u: $Qf(u)$; already derived on right hand side of Eq. (4);
- Novelty factor of User Story u: $Nf(u)$;
- Type factor of User Story u: $Tf(u)$.

First term on right hand side of Eq. (5) is estimated Story Points of User Story u without inclusion of risk factor. It is given as,

$$esp(u) = Sf(u) \times Cf(u) + Qf(u) \times Nf(u) \times Tf(u). \tag{8}$$

Expressions in Eqs. (8) and (6) complete the derivation of Eq. (5). Now estimated Story Points for Sprint s, $ESPS(s)$ is given as,

$$ESPS(s) = \sum_{u \in S(s)} ESP(u), \tag{9}$$

where $S(s)$ is the set of User Stories in Sprint s. Effort for the project is the sum of efforts of all Sprints. This concludes derivation of our risk based effort estimation model.

4. RESULTS AND DISCUSSION

In this section, we test the accuracy of our model using various performance metrics and compare it with other works. To this end, we have developed an in-house simulator to create project scenarios. Each project evolves as a result of multiple Sprints each one of which has User Stories that are designed and implemented by various distributed teams. In general, progress of teams improves with Sprints due to increasing knowledge about project relevant artifacts. Progress can be measured by the number of User Stories completed in each Sprint. Any effort estimation method should preferably predict only for the next Sprint. Performance feedback on the previous Sprint gives valuable information about User Stories requirements that depend on a hoard of factors as discussed earlier and actual capabilities to address those requirements. Thus, accuracy of effort estimation also increases with Sprints.

For our mobile application project scenario, we assume three teams to create a distributed development setup: Lahore team, Dubai team and New York (NY) team. Other parameters are generated randomly. There are 128 User Stories in the project that are allocated to teams arbitrarily but randomly from Uniform Distribution. User Stories are indexed sequentially per Sprint for better understandability. There are three types of data used to derive results:

1. Using our proposed model, Story Point estimations without considering the risk factor. For keeping brevity in results, it is denoted by PM and refers to Eq. (8). Let $ST(s, t)$ denote the set of User Stories executed by team t, $t \in$ {Lahore, Dubai, NY} in Sprint s, then $PM(s, t) = \sum_{u \in ST(s,t)} esp(u)$. Since data in tables has a grid view, hence notation PM only suffices.
2. Using our proposed model, Story Point estimations with inclusion of risk factor. It is

denoted by PMr and refers to Eq. (5). Now PMr(s, t) = $\sum_{u \in ST(s,t)} ESP(u)$. Here notation PMr only suffices.

3. Actual Story Points refer to the amount of work completed per Sprint per team. It is denoted as 'Actual' in results.

For each User Story, the considered factors (Size, Complexity, Quality, Novelty, Type and Risk) obtain random values from Normal Distributions (NDs) as follows:

- Estimated Story Points for PM are based on ND with Mean value 20% more than the middle value of the level/value chosen for the tagged factor and Standard Deviation 10% of possible range. Here risk is not considered.
- Estimated Story Points for PMr are derived from ND with restrictions similar to PM, except that risk is considered;
- With respect to the Sprint estimations, the actual amount of work completed in each Sprint is recorded for each team in the form of Actual Story Points. They are drawn from ND with Mean value 30% more than the middle value of the level/value chosen for the tagged factor and Standard Deviation 5% of the possible range;. Allowing higher values for completed Story Points reflect more uncertainty due to actual conditions encountered during project development.

Allocation to User Stories to various distributed teams during various Sprints, and corresponding PM/PMr Story Points estimations and Actual Story Points completed are listed in Table 7. Understandably estimations have a disparity with actual work completed. Disparity is a valuable signal about the prediction accuracy of system factors. Estimations for subsequent Sprint consider this disparity with an aim to reduce it. Mostly disparity appears in the form of incompleteness of User Stories in the actual executions. We assume that User Stories are executed sequentially, so that incompleteness, if any, occurs only for the high indexed User Stories for the concerned Sprint. We represent this incompleteness by introducing few notations that are used in Table 7.

- If a User Story is not completed in the current Sprint, it is carried to the next Sprint. Such execution in two Sprints is shown as <<n>> in both Sprints, where n is the User Story index;
- Despite predicted execution of a User Story in the current Sprint, if it could not even be started, it is carried to the next Sprint entirely, where it is shown as <n>;
- Last User Story of last Sprint can be completed before the actual completion of project. This state is shown as >>n<<.

Data in Table 7 is plotted in Fig. 1 with bar labels showing the User Stories per Sprint per team. It can be observed clearly that the disparity of both estimations, PM and PMr from actual productivity is decreasing with Sprints. This owes to the fact that Agile based development emphasizes on adjustment of influencing factors during the span of project rather than following some development process rigidly. Due to such flexibility, actual progress during previous Sprints is considered before making new predictions. Hence the said disparity tends to decrease with Sprints. Another observation is increasing amount of productivity with Sprints. This owes to the increasing awareness of teams as the project progresses about the relevant artifacts.

Table 7. Proposed Model (PM and PMr): a comparison between predicted Story Points vs Actual Story Points completed.

Sprint			1			2			3			4			5			6		
Team			Lahore	Dubai	NY	Lahore	Dubai	NY	Lahore	Dubai	NY	Lahore	Dubai	NY	Lahore	Dubai	NY	Lahore	Dubai	NY
Story Points			136	98	96	135	98	95	134	96	93	133	94	94	131	93	92	133	96	96
			117	85	87	117	87	87	120	90	90	124	90	90	131	92	92	132	95	92
		Actual	108	80	83	111	81	85	116	86	87	122	90	91	131	91	93	131	94	92
User Stories		PM	1, 2, 3, 4, 5, 6, 7, 8, 9	10, 11, 12, 13, 14, 15, 16, 17, 18, <<19>>	20, 21, 22, 23, 24, 25, <<26>>	<<8>>, <9>, 27, 28, 29, 30, 31, 32, 33	<<18>>, <19>, 34, 35, 36, 37, 38, <<39>>	<<25>>, <26>, 40, 41, 42, 43, 44, 45, <<46>>	<<32>>, <33>, 47, 48, 49, 50, 51, 52, 53, <<54>>	<<37>>, <38>, <39>, 55, 56, 57, 58, 59, <<60>>	<<45>>, <46>, 61, 62, 63, 64, 65, <<66>>	<<53>>, <54>, 67, 68, 69, 70, 71, 72, 73, <<74>>	<<59>>, <60>, 75, 76, 77, 78, 79	<<65>>, <66>, 80, 81, 82, 83, 84, <<85>>	<<74>>, 86, 87, 88, 89, 90, 91, 92, 93, <<94>>	<<79>>, 95, 96, 97, 98, 99, 100, <<101>>	<<85>>, 102, 103, 104, 105, 106, <<107>>	<<94>>, 108, 109, 110, 111, 112, 113, 114, 115, >>116<<	<<94>>, 108, <101>, 117, 118, 119, 120, 121, >>122<<	<<107>>, 123, 124, 125, 126, 127, >>128<<
		PMr	1, 2, 3, 4, 5, 6, 7, 8	10, 11, 12, 13, 14, 15, 16, 17, <<18>>	20, 21, 22, 23, 24, 25	<<8>>, 9, 27, 28, 29, 30, 31, <<32>>	<<18>>, 19, 34, 35, 36, 37, <<38>>	<<25>>, 26, 40, 41, 42, 43, 44, 45	<<32>>, 33, 47, 48, 49, 50, 51, 52, <<53>>	<<37>>, <38>, 39, 55, 56, 57, 58, 59, <<60>>	<<45>>, 46, 61, 62, 63, 64, 65, <<66>>	<<53>>, 54, 67, 68, 69, 70, 71, 72, 73, <<74>>	<<59>>, <60>, 75, 76, 77, 78, <<79>>	<<65>>, <66>, 80, 81, 82, 83, 84, <<85>>	<<74>>, 86, 87, 88, 89, 90, 91, 92, 93, <<94>>	<<79>>, 95, 96, 97, 98, 99, 100, <<101>>	<<85>>, 102, 103, 104, 105, 106, <<107>>	<<94>>, 108, 109, 110, 111, 112, 113, 114, 115, 116	<<94>>, 108, <101>, 117, 118, 119, 120, 121, 122	<<107>>, 123, 124, 125, 126, 127, >>128<<
	Actual		1, 2, 3, 4, 5, 6, 7, <<8>>	10, 11, 12, 13, 14, 15, 16, 17, <<18>>	20, 21, 22, 23, 24, <<25>>	<<8>>, 9, 27, 83, 29, 30, 31, <<32>>	<<18>>, 19, 34, 35, 36, <<37>>	<<25>>, 26, 40, 41, 42, 43, 44, <<45>>	<<32>>, 33, 47, 48, 49, 50, 51, 52, <<53>>	<<37>>, 38, 39, 55, 56, 57, 58, <<59>>	<<45>>, 46, 61, 62, 63, 64, <<65>>	<<53>>, 54, 67, 68, 69, 70, 71, 72, 73, <<74>>	<<59>>, 60, 75, 76, 77, 78, <<79>>	<<65>>, 66, 80, 81, 82, 83, 84, <<85>>	<<74>>, 86, 87, 88, 89, 90, 91, 92, 93, <<94>>	<<79>>, 95, 96, 97, 98, 99, 100	<<85>>, 102, 103, 104, 105, 106, <<107>>	<<94>>, 108, 109, 110, 111, 112, 113, 114, 115, >>116<<	<<94>>, 108, 101, 117, 118, 119, 120, 121, 122	<<107>>, 123, 124, 125, 126, 127, >>128<<

In order to elucidate the disparity issue, we calculate Mean Relative Error (MRE) [33] of estimated Story Points (PM/PMr) at Sprint level per team as,

$$MRE = \frac{|\text{Actual StoryPoint} - \text{PMr Story Points}|}{\text{Actual Story Points}} \times 100. \tag{10}$$

MRE is plotted in Fig. 2, where it is observed to be decreasing with Sprints in both estimations, PM and PMr. We have also captured MRE decrease trend through fitting of degree 3 polynomials. There is a slight rise of MRE at the end in case of PM. This is due to the fact that PM is risk agnostic, hence does not benefit from better estimation of risks as Sprints occur.

To investigate whether the data trend presented in Fig. 2 has team specificity, team transparent MRE data trend in Fig. 3 is plotted by taking team averages. Again data trend is captured through fitting of degree 3 polynomials. In principle, data trend of Fig. 3 confirms the observations made for data trend of Fig. 2.

MRE is based on individual comparison; any skewed data may go unnoticed. This deficiency can be overcome by determining prediction accuracy, PRED(x) [33]:

$$PRED(x) = \frac{j}{k} \times 100, \tag{11}$$

where j is count of MRE values that are at most x% and k is total count of MRE values. A plot is given in Fig. 4 for $0 \leq x \leq 100$. Both plots tend to attain 100% accuracy at x = 30%. MRE values of PMr are more clustered at low end than PM. Though PRED(x) is more meaningful than MRE but establishing it as a prediction indicator is only based on the realm that results from a controlled experiment generate 'clean' or 'sanitized' data [33]. Accuracy level reflects more on how well the data generating process is followed. Finally results from our Proposed Model (PMr) are compared with those from existing works. Basically, not a single reference is found that presents results for our context of DASD. Nevertheless we include results for ASD for comparison (see Table 8). In [9], experts and students use Planning Poker to estimate efforts for given scenarios. Due to more domain knowledge and experience in using Planning Poker, experts achieve more accuracy. The disparity in the estimations of students is lessened when statistical techniques are used that smooth high fluctuations in student estimations. Balanced measure of relative error (BRE) is given for experts. Comparing these values with BRE of PMr indicates relatively better values of the latter. Partially it is due to inclusion of risk factor in our Proposed Model and partially due to deriving our results from simulations, which is a controlled environment.

Different Neural Networks such as General Regression Neural Network (GRNN), Probabilistic Neural Network, GMDH Polynomial Neural Network and Cascade Correlation Neural Network (CCNN) are used for Agile Software effort estimation based on Story Points [25]. Obtained results are empirically validated and compared. CCNN outperforms others in terms of MMRE (=0.148) and PRED(x) (=94.76%), while GRNN gives worst results. MMRE is computed as the average of all MRE values. PMr gives better results than CCNN in terms of both MMRE (=0.0044) and PRED(x) (=93.51%). This observation is due to two reasons: one that, though there are many over estimations

in PMr due to inclusion of risk factor but the extent of over estimations is relatively small, hence MMRE, which is a cumulative measure, is low; two that, our over estimations are clustered on lower end, thus comparisons on PRED(25) or PRED(30) are better in our case.

Effort estimation using Machine Learning (ML) for different projects running in Capability Maturity Model Integration (CMMI) organizations are presented in [23]. Here estimation accuracy due to ML improves over a method that estimates effort as a product of estimated size of project and productivity factor, so that MER decreases (for size, from 28% to 5.9% and for effort, from 34% to 12%). Effort estimation by PMr reduces MER to 34.33% (from 8.62% to 2.96%) as compared to 35.29% reduction by ML (from 34% to 12%).

Effort estimation is done using a method, J48 + Planning Poker. In this method Planning Poker is augmented by ML. Results are presented in terms of number of correct/over/under estimations. When PMr is compared on these metrics, it sways high for over estimations. This is understandable by considering the risk factor. We consider that comparing results only on these metrics is quite insufficient as the extent of errors is not captured.

5. CONCLUSIONS

Software development trends aim at benefiting from Agile approach and distributed setups. Being emergent fields, decision making in both dimensions is a hard task unless more knowledge about artifacts is established. The challenge is many folds when predictability on effort is required for better deployment of resources and preparedness beforehand. This paper addresses this challenge for mobile application development scenario. We formally express our main research question in Eq. (1) that expresses dependency of effort on various influential factors, and solve it as Eq. (5).

We have keenly identified and focused on issues that may arise due to distributed setups using Agile approach. There are three contributions: one, quality aspect of User Stories is introduced to be able to tradeoff between quality and effort. To achieve higher quality of development, more effort or cost or both are required. This tradeoff offers appropriate deployment of resources for time constrained projects. Two, novelty and type factors are considered to represent emerging application domains such as mobile related. Three, risk factor is included in its own merit to accommodate uncertainty that may occur during project development.

Our proposed model is verified by an in-house simulator that is capable to generate project scenarios. For each scenario, estimations for effort with(out) risk are made, against which actual effort needed is compared. Such an approach allows requirements management towards a common goal of more precise predictions for future projects. Our model supports effort estimation for DASD and serves as a basis for further investigations.

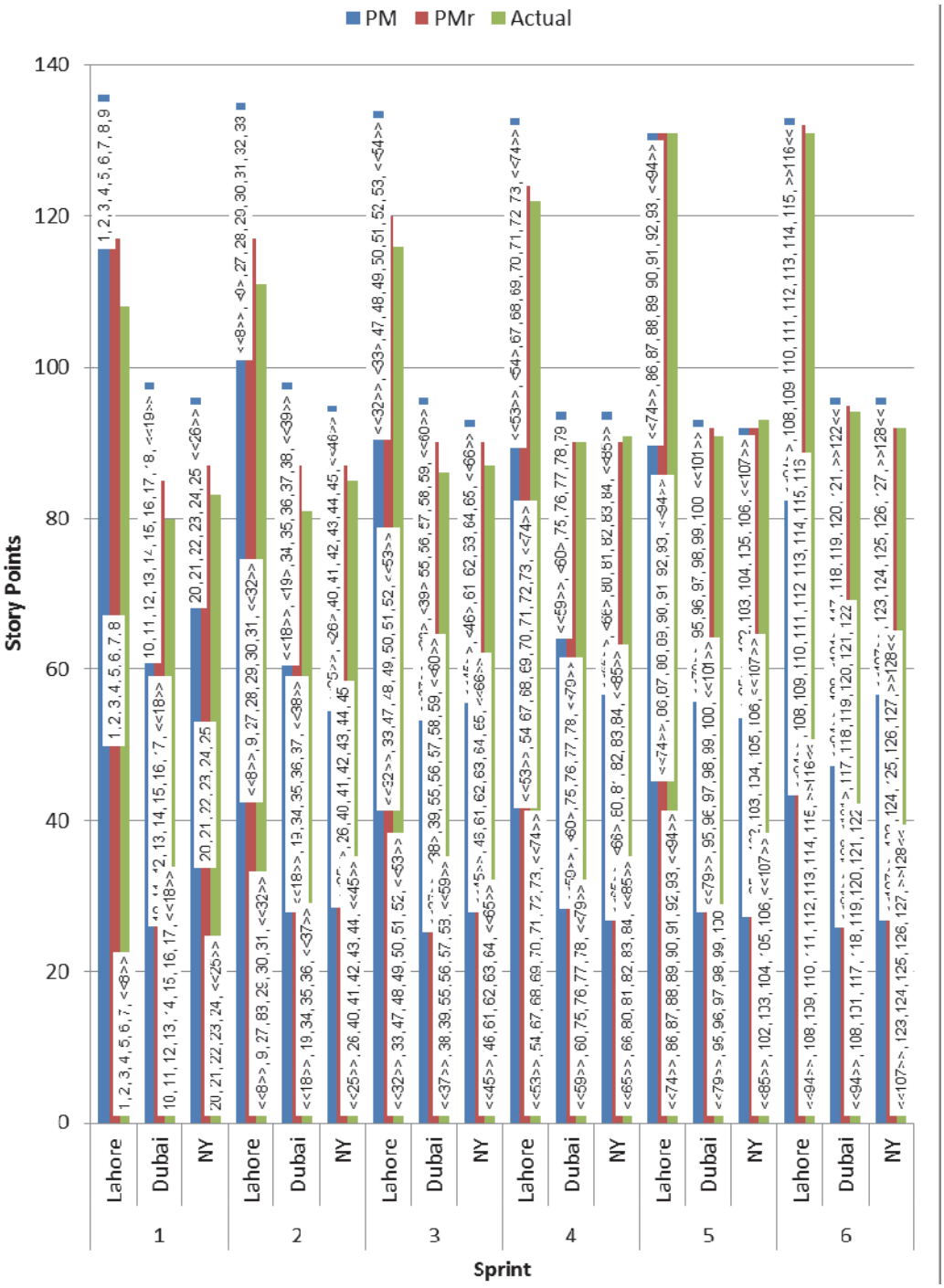

Fig. 1. A comparison between our Proposed Model (with/without risk factor inclusion) predicted Story Points vs Actual Story Points completed. Bar labels show the User Stories executed per Sprint per team.

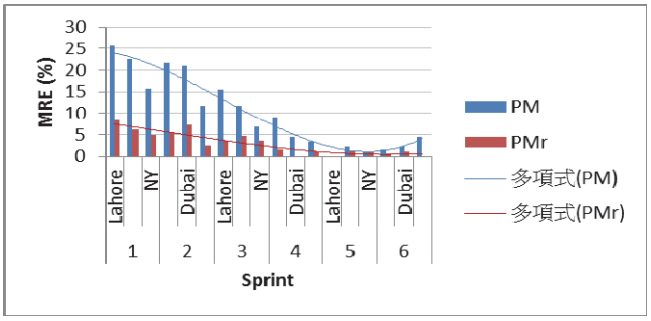

Fig. 2. MRE (%) of both estimations (PM/PMr) seen decreasing with Sprints. Decrease trends are well represented by polynomials of degree 3 in each case.

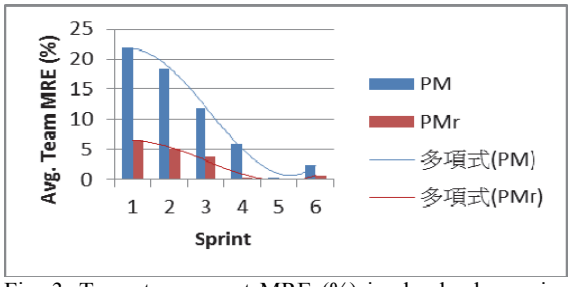

Fig. 3. Team transparent MRE (%) is clearly decreasing with Sprints. In this case also, polynomials of degree 3 fit the data well.

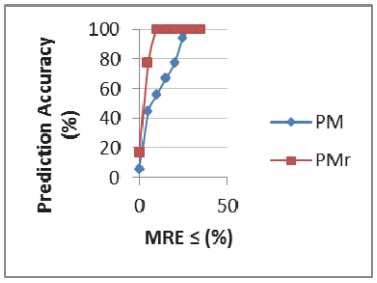

Fig. 4. Prediction Accuracy gives a collective view of all MRE values.

Table 8. Our proposed model vs existing models: a comparison on various metrics.

Method	MER	MMRE	PRED (%)	BRE						Correct Est. (%)	Over Est. (%)	Under Est. (%)
				Mean	Median	SD	Skewness	Kurtosis	Range			
Planning Poker [9]	×	×	×	0.428	0.326	0.396	1.694	2.666	1.40	×	×	×
Cascade Correlation Neural Network [25]	×	0.148	94.76	×	×	×	×	×	×	×	×	×
Machine Learning Model [23]	34% to12% (decreases)	×	×	×	×	×	×	×	×	×	×	×
J48 + Planning Poker [26]	×	×	×	×	×	×	×	×	×	31	34.9	34.1
Proposed Method **(PMr)**	8.62% to 2.96% (decreases)	0.0044	93.51	0.031	0.023	0.029	1.600	2.918	0.118	16.667	72.222	11.111

REFERENCES

1. S. Abolfazli, Z. Sanaei, A. Gani, F. Xia, and L. Yang, "Rich mobile applications: genesis, taxonomy, and open issues," *Journal of Network and Computer Applications*, Vol. 40, 2014, pp. 345-362.
2. A. I. Wasserman, "Software engineering issues for mobile application development," in *Proceedings of ACM FSE/SDP Workshop on Future of Software Engineering*

Research, 2010, pp. 397-400.
3. R. Britto, E. Mendes, and J. Börstler, "An empirical investigation on effort estimation in agile global software development," in *Proceedings of the 10th IEEE International Conference on Global Software Engineering*, 2015, pp. 38-45.
4. S. V. Shrivastava and U. Rathod, "Risks in distributed agile development: A review," *Procedia-Social and Behavioral Sciences*, Vol. 133, 2014, pp. 417-424.
5. A. Santos, J. Kroll, A. Sales, P. Fernandes, and D. Wildt, "Investigating the adoption of agile practices in mobile application development," in *Proceedings of the 18th International Conference on Enterprise Information Systems*, Vol. 1, 2016, pp. 490-497.
6. P. Abrahamsson, A. Hanhineva, H. Hulkko, T. Ihme, J. Jäälinoja, M. Korkala, J. Koskela, P. Kyllönen, and O. Salo, "Mobile-D: an agile approach for mobile application development," in *Proceedings of the 19th Annual ACM Sigplan Conference on Object-Oriented Programming*, *Systems*, *Languages*, *and Applications*, 2004, pp. 174-175.
7. Q. Lu, T. Wu, J. Yan, J. Yan, F. Ma, and F. Zhang, "Lightweight method-level energy consumption estimation for Android applications," in *Proceedings of the 10th International Symposium on Theoretical Aspects of Software Engineering*, 2016, pp. 144-151.
8. H. Zahraoui and M. A. Janati Idrissi, "Adjusting story points calculation in scrum effort & time estimation," in *Proceedings of the 10th International Conference on Intelligent Systems: Theories and Applications*, 2015, pp. 1-8.
9. V. Mahnič and T. Hovelja, "On using planning poker for estimating user stories," *Journal of Systems and Software*, Vol. 85, 2012, pp. 2086-2095.
10. M. Usman, E. Mendes, and J. Börstler, "Effort estimation in agile software development: a survey on the state of the practice," in *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*, 2015, pp. 12:1-12:10.
11. D. A. Montini, D. Battaglia, G. R. Matuck, A. M. d. Cunha, L. A. V. Dias, A. A. Montini, and B. Shahzad, "A chi-square methodology applied in deviations control of project plan to support the RIMAM model," in *Proceedings of the 11th International Conference on Information Technology: New Generations*, 2014, pp. 9-14.
12. B. Shahzad and A. M. Said, "Identification and quantitative analysis of project success factors for large scale projects," *International Journal of Knowledge Society Research*, Vol. 5, 2014, pp. 83-95.
13. S. K. T. Ziauddin and S. Zia, "An effort estimation model for agile software development," *Advances in Computer Science and its Applications*, Vol. 314, 2012, pp. 314-324.
14. B. Shahzad and A. S. Al-Mudimigh, "Risk identification, mitigation and avoidance model for handling software risk," in *Proceedings of IEEE International Conference on Computational Intelligence*, *Communication Systems and Networks*, 2010, pp. 191-196.
15. E. Coelho and A. Basu, "Effort estimation in agile software development using story points," *International Journal of Applied Information Systems*, Vol. 3, 2012, pp. 7-10.
S. Kang, O. Choi, and J. Baik, "Model-based dynamic cost estimation and tracking method for agile software development," in *Proceedings of the 9th IEEE/ACIS International Conference on Computer and Information Science*, 2010, pp. 743-748.

16. M. R. Braga, C. I. Bezerra, J. M. S. Monteiro, and R. Andrade, "A pattern language for agile software estimation," in *Proceedings of the 9th ACM Latin-American Conference on Pattern Languages of Programming*, 2012, pp. 1-15.
17. R. Litoriya and A. Kothari, "An efficient approach for agile web based project estimation: AgileMOW," *Journal of Software Engineering and Applications*, Vol. 6, 2013, pp. 297-303.
18. F. Raith, I. Richter, R. Lindermeier, and G. Klinker, "Identification of inaccurate effort estimates in agile software development," in *Proceedings of the 20th Asia-Pacific Software Engineerning Conference*, 2013, pp. 67-72.
19. A. W. M. M. Parvez, "Efficiency factor and risk factor based user case point test effort estimation model compatible with agile software development," in *Proceedings of International Conference on Information Technology and Electrical Engineering*, 2013, pp. 113-118.
20. R. Popli and N. Chauhan, "Agile estimation using people and project related factors," in *Proceedings of International Conference on Computing for Sustainable Global Development*, 2014, pp. 564-569.
21. I. Manga and N. Blamah, "A particle swarm optimization-based framework for agile software effort estimation," *The International Journal Of Engineering And Science*, Vol. 3, 2014, pp. 30-36.
22. A. E. D. Hamouda, "Using agile story points as an estimation technique in CMMI organizations," in *Proceedings of IEEE Agile Conference*, 2014, pp. 16-23.
23. R. Popli and N. Chauhan, "Cost and effort estimation in agile software development," in *Proceedings of International Conference on Reliability Optimization and Information Technology*, 2014, pp. 57-61.
24. A. Panda, S. M. Satapathy, and S. K. Rath, "Empirical validation of neural network models for agile software effort estimation based on story points," *Procedia Computer Science*, Vol. 57, 2015, pp. 772-781.
25. K. Moharreri, A. V. Sapre, J. Ramanathan, and R. Ramnath, "Cost-effective supervised learning models for software effort estimation in agile environments," in *Proceedings of the 40th IEEE Annual Computer Software and Applications Conference*, 2016, pp. 135-140.
26. R. Britto, V. Freitas, E. Mendes, and M. Usman, "Effort estimation in global software development: A systematic literature review," in *Proceedings of the 9th IEEE International Conference on Global Software Engineering*, 2014, pp. 135-144.
27. S. V. Shrivastava and U. Rathod, "Categorization of risk factors for distributed agile projects," *Information and Software Technology*, Vol. 58, 2015, pp. 373-387.
28. S. Sundararajan, M. Bhasi, and P. K. Vijayaraghavan, "Case study on risk management practice in large offshore-outsourced agile software projects," *IET Software*, Vol. 8, 2014, pp. 245-257.
29. B. C. Rajan and M. S. Kumar, "Contribution of risk assessment techniques in effort estimation of software development process," *PARIPEX-Indian Journal of Research*, Vol. 5, 2016, pp. 498-499.
30. ISO/IEC 25010:2011, Systems and Software Engineering, Systems and Software Quality Requirements and Evaluation, System and Software Quality Models, ISO, 2011.
31. A. Spriestersbach and T. Springer, "Quality attributes in mobile web application

development," F. Bomarius, H. Iida, eds., *Product Focused Software Process Improvement*, LNCS Vol. 3009, 2004, pp. 120-130.
32. M. Korte and D. Port, "Confidence in software cost estimation results based on MMRE and PRED," in *Proceedings of the 4th International Workshop on Predictor Models in Software Engineering*, 2008, pp. 63-70.

Waqar Aslam obtained the HEC Oversees Scholarship to receive his Ph.D. degree in Computer Science from The Eindhoven University of Technology, The Netherlands. He is an Assistant Professor at the Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Pakistan. His research interests include performance modeling of (distributed) software architectures, effort/time/cost estimation of software development in (distributed) agile setups, performance modeling and QoS of wireless/computer networks and social network data analysis.

Farah Ijaz received the MS(CS) degree from Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Pakistan. She is a visiting Lecturer at the same department, where she is also doing research. Her research interests include software requirement management in distributed environment using agile methods with focus on effort/ time/cost estimation.

M. IkramUllah Lali is working as an Associate Professor at the Department of SE, University of Gujrat, Pakistan. He obtained Masters in Software Engineering and Ph.D. Computer Science degrees from COMSATS, Islamabad, Pakistan. His areas of interests include machine learning, social network data analysis, formal methods and software testing. He is author of more than 40 research articles which have been published in conferences and reputed journals.

Waqar Mehmood obtained the HEC Oversees scholarship to receive his Ph.D. degree in Computer Science (Software Engineering) in 2011 from University of Insbruck, Austria. He obtained CUST 100% fee waiver scholarship to receive his Master degree in Software Engineering from CUST, Islamabad. Currently he is working as an Assistant Professor at COMSATS Institute of Information Technology, Wah Cantt, Pakistan. His research interests include model driven engineering, software configuration management and software testing.