Developing and Using Checklists to Improve Software Effort Estimation: a Multi-Case Study

Muhammad Usman^{a,*}, Kai Petersen^a, Jürgen Börstler^a, Pedro Santos Neto^b

^aDepartment of Software Engineering, Blekinge Institute of Technology, 371 79, Karlskrona, Sweden

^bFederal University of Piauí, Teresina, Brazil

Abstract

Expert judgment based effort estimation techniques are widely used for estimating software effort. In the absence of process support, experts may overlook important factors during estimation, leading to inconsistent estimates. This might cause underestimation, which is a common problem in software projects. This multi-case study aims to improve expert estimation of software development effort. Our goal is two-fold: 1) to propose a process to develop and evolve estimation checklists for agile teams, and 2) to evaluate the usefulness of the checklists in improving expert estimation processes. The use of checklists improved the accuracy of the estimates in two case companies. In particular, the underestimation bias was reduced to a large extent. For the third case, we could not perform a similar analysis, due to the unavailability of historical data. However, when checklist was used in two sprints, the estimates were quite accurate (median Balanced Relative Error (BRE) bias of -0.05). The study participants from the case companies observed several benefits of using the checklists during estimation, such as increased confidence in estimates, improved consistency due to help in recalling relevant factors, more objectivity in the process, improved understanding of the tasks being estimated, and reduced chances of missing tasks.

Keywords:

Expert judgment based effort estimation, checklist, agile software development, case study

1. Introduction

Effort estimation techniques based on expert judgment are the most commonly used techniques for estimating software development effort [1, 2, 3]. In expert estimation, the involved experts rely on their knowledge and previous experience to subjectively analyze a number of factors to estimate the development effort. Experts perform this subjective assessment using their “intuition”, which is described as a “non-explicit and non-recoverable reasoning process” by Jørgensen [2]. In some cases experts may rely on their intuition only, while in others experts may use historical data and/or checklists, subject to their availability, to make more informed estimates.

Expert estimation involves little documentation and a light-weight process. Compared to algorithmic estimation models, such as COCOMO I and II [4, 5], the focus is on the experts performing the estimation. These attributes of expert estimation synchronize well with agile processes, where individuals and their interactions are valued more than processes and tools.

The lack of formalism in expert estimation may have some negative consequences; in the absence of any explicit structure, expert estimation can become an ad-hoc activity. Furthermore, experts may miss important activities and tasks (e.g., testing effort [3] or non-functional requirements [3, 6]) and as a result underestimate development effort. In early project stages uncertainty is also high, e.g., due to incomplete and changing requirements, which may lead to inconsistent expert estimates [7]. Experts are normally aware of the factors that should be considered. However,

*Corresponding author

Email addresses: muhammad.usman@bth.se (Muhammad Usman), kai.petersen@bth.se (Kai Petersen), jurgen.borstler@bth.se (Jürgen Börstler), pasn@ufpi.edu.br (Pedro Santos Neto)

they might not always recall and consider these factors consistently during estimation. In the agile context, where team-based expert estimation is the norm [8, 3] and there is lack of documentation [9], junior or new team members might not be aware of all activities and factors that should be accounted for during effort estimation. Furthermore, relatively less emphasis on documentation in the agile context is also likely to result in the loss of the useful estimation experience generated during the effort estimation process.

Checklists have the potential to support estimation in addressing the issues described above [2]. Checklists document the factors and activities that should be considered during effort estimation, which ensures that they are taken into account. Checklists are also useful in supporting new team members [2], who would be less aware of the activities and factors that should be considered during estimation. Checklists can also prove helpful in documenting and reusing estimation data and experience to improve estimation accuracy over time [10].

There is evidence that the use of checklists helps to improve estimation accuracy (e.g., [11, 12]). However, we have not seen any empirical study in software engineering that describes and demonstrates how to develop and evolve such checklists for agile teams. The present study addresses these gaps by making the following contributions:

- Propose a process to develop and evolve a customized checklist to improve expert estimation.
- Apply the process with agile teams in three different companies to develop and evolve checklists for their specific contexts.
- Evaluate the usefulness of the proposed checklists using both qualitative and quantitative data.

The remainder of the paper is organized as follows: Section 2 details the background of this work and presents related work on the use of checklists in software effort estimation; the research methodology is described in Section 3; sections 4 and 5 present the results and analysis; Section 6 provides further discussions on these results; Section 7 describes validity threats, while Section 8 concludes the paper.

2. Background and related work

In this section first we discuss the use of checklists in other disciplines such as aviation and health care. Next, we briefly discuss use of checklists in software engineering in general, also in effort estimation in particular. Lastly, we describe our previous works that provided important inputs and motivation for the work presented in this study.

2.1. Checklists in other disciplines

Checklists are widely used to avoid human error, improve safety and performance in variety of disciplines, such as health care, aviation and product manufacturing [13]. In complex environments, experts are likely to overlook routine matters due to fallible memory and distraction under pressing situations [14]. Furthermore, experts can knowingly skip certain tasks with the assumption that certain tasks are not always required. Checklists have the potential to help avoiding such lapses [14]. They are widely used in fields, such as aviation and health care, where safety and accuracy are critical to reduce human error [13].

Checklists became a standard operating procedure in the aviation industry after the crash of the Boeing¹ Model 299 during the trial demonstration in 1935 [15]. From mid 1990s, electronic checklists were introduced in aircrafts to prevent crew errors associated with paper checklists [16, 17].

Pronovost proposed and used a simple checklist consisting of five practices to avoid human mistakes, which were responsible for causing deadly infections to patients in surgical Intensive Care Units (ICUs) [18]. The intervention resulted in a large reduction (up to 66%) in infection rates [19]. Later, Dr. Pronovost's idea was adopted by all other states of the USA [20], and was also replicated at other places (cf. [21]).

¹<http://www.boeing.com/>

2.2. Checklists in software engineering

Checklist have long been used as an aid during inspections of different software artifacts such as requirements and design documents, and code [22]. Checklists support inspectors or reviewers in systematically evaluating or reading the software artifact being inspected (cf. [23, 24, 25, 26]). Checklists are also used in empirical software engineering research (cf. [27, 28]) to assess the quality of the empirical studies. These inspection and review checklists can be classified as “evaluative checklists” [29], as the objective is to evaluate or review an artifact or document. Evaluation checklists improve the consistency of the evaluations by providing reviewers a set of practices or guidelines or criteria to be used during evaluation [30, 29].

Besides evaluation, checklists are also employed as mnemonic instruments that serve as a “reminder system” to help experts in consistently applying procedures and processes [29]. Mnemonic checklists facilitate experts in remembering the relevant information and steps, which they can otherwise forget due to lapse in attention or memory [14]. In software effort or cost estimation, such checklists have been used to support estimators during the estimation activity [2]. Jørgensen [2] noted that the checklists are beneficial as they support experts in not forgetting relevant activities and factors during estimation, increase their confidence in estimates and encourage them to use more accurate strategies during estimation. Similar results were found by Passing and Shepperd [12], who performed an experiment with student subjects to investigate how the use of checklists and group discussions support expert estimation. They noted that the checklists improved the consistency and transparency of size and effort estimates. Further, they also found that the use of checklist increased the subjects’ confidence in their estimates. The checklist helped student estimators in recalling the relevant factors that should be considered during estimation.

Furulund and Moløkken-Østvold [11] found that projects where checklists were used during estimation have relatively accurate estimates as compared to the projects where checklists were not used.

Jørgensen [31] proposed a software cost management process framework and a preliminary checklist that follows the structure of the framework. The framework consists of four phases and several activities. The four phases are: preparation, estimation, application and learning phases. The framework and accompanying checklist have a broad scope, as they go beyond the typical estimation activity. Our goal is to propose and demonstrate a process to develop and evolve customized checklists specifically for the estimation activity to support practitioners in arriving at better estimates in an agile context.

2.3. Process support for software estimation checklists

There is a lack of process support for developing and implementing software estimation checklists. In the study by Furulund and Moløkken-Østvold [11], the focus is on demonstrating the usefulness of checklists and experience data in increasing the estimation accuracy. How the checklist was developed, evolved and used during effort estimation was not discussed. Likewise, Passing and Shepperd [12] investigated the use of checklist and group discussions in supporting expert estimation. In this study as well, the focus was not on process support for developing and implementing the checklist. The checklist contents are listed and consist of the typical activities (e.g., manage requirements, class design, prototype development, defect fixing, unit testing, documentation etc.) in a development project.

As mentioned previously, the preliminary checklist proposed by Jørgensen [31] is structured on a project management framework consisting of four phases and twelve activities. The checklist is based on a number of sources, including estimation checklists from the six Norwegian software companies, and the literature on cost management, forecasting and estimation. The scope of the checklist is much wider than the typical estimation activity, and also includes phases, such as preparation phase, beyond the typical estimation activity. In the context of the agile teams, wherein estimation is performed repeatedly with each release and iteration, it may not be feasible to use a long checklist covering activities beyond the typical estimation activity. Furthermore, checklist factors should be empirically grounded in the context in which they are intended to be used. Jørgensen [31] suggested that the proposed preliminary checklist should be customized to include only the relevant issues.

Hales et al. [29] recommended that the contents of the checklist should be identified systematically through a review of the peer-reviewed literature, and should also take into account the contextual factors of the local hospital. Our aim is also to develop estimation checklists, and accompanying process support, that are based both on published literature and also on the systematically elicited input from the involved agile teams. Hales et al. [29] also suggested that the checklist should not be onerous, and should first be pilot tested in a controlled environment. We also aim to develop a customized and light-weight checklist. Furthermore, before transferring the checklist to the involved teams,

we aim to present, validate and revise it first in a controlled setting, and then later evolve it further based on feedback from the actual use.

2.4. Agile estimation taxonomy

We organized the state of the art and practice on effort estimation in agile context as a facet based taxonomy [32], referred here as AET (Agile Estimation Taxonomy). At the top level, AET has four dimensions: context, estimation technique, effort predictors and effort estimate. Software effort estimation is carried out in a certain **context** (e.g., company/team size, domain, agile practices followed etc.) in which the estimator(s) use some **estimation technique (s)** (e.g., expert judgment, case base reasoning etc.) and **effort predictors** (e.g., size, teams expertise etc.) to arrive at an **effort estimate** (e.g., point estimate in hour or three point estimate etc.). These dimensions in turn have several facets, which were identified through a systematic review [8] and survey [3] on effort estimation in agile context. We used AET to characterize the effort estimation process of the three cases included in this study in Section 3.

We used AET to characterize effort estimation processes of three companies. These companies found AET useful in documenting their effort estimation sessions [32]. The AET, however, was not used during the effort estimation process. To do so, we decided to develop a checklist instrument to support and structure the otherwise largely ad-hoc expert judgment based effort estimation process, which is the most frequently practiced estimation technique in software industry [2, 33].

The factors identified in AET served as important input in this study. In particular, AET's effort predictor dimension encompasses factors (see Table 1) that estimators consider during the effort estimation process. Further input to this study came from an industrial case study on effort estimation in large scale development [34]. We investigated and identified factors (Table 1) that impact the accuracy of the effort estimates. Based on the data and participants' opinion, the study recommended that it is important to consider the identified factors during effort estimation, as they can potentially lead to effort overruns in the project. Table 1 lists factors identified in our previous works, which acted as a starting point in this study.

Table 1: Potential checklist factors from previous works

Factor	AET	Case Study
Product size	✓	✓
Team's skill level	✓	✓
Multi-site development issues	✓	✓
Team's prior experience with similar tasks	✓	-
Non-functional requirements	✓	-
Customer communication	✓	-
Customer priority	-	✓

3. Research methodology

We selected case study methodology to investigate effort estimation, which is the phenomenon under study. Based on Runeson et al.'s classification [27], it can be classified as "improving" case study as our goal was to improve the expert effort estimation process.

3.1. Research context

The study is performed in three different case companies. The details about these companies, their products and processes are described in the following.

3.1.1. Case company 1: Infoway

Infoway² is a medium-sized Brazilian company developing software applications for the health sector. It has over 40 employees in different development roles. We worked with two teams working on two different products at Infoway: Product A and B.

²<http://infoway-br.com/>

Product A is a management information system for health insurance providers. The product supports insurance providers in marketing and sales management, order management, customer records and payment management. It also provides an interface to hospitals through which they can register patients and interact with their corresponding insurance providers for authorization of the treatments. The development team consists of three developers, one team lead and a project manager. The product has been evolved for over 7 years, and is being used by many insurance companies in Brazil. The product is being developed in Java³ and consists of approximately 0.41 million lines of code.

Product B is a business intelligence application to support health insurance providers to optimize their insurance plans. It helps the insurance providers to setup key indicators to monitor the pattern of physicians recommending different examinations to their customers. The product notes and reports the unusual behavior/pattern as an alert to the managers of the insurance provider. The managers can set up new indicators, and also define/change the thresholds for alerts associated with different indicators. The development team consists of five developers and a team lead (who also performs the project management tasks). The product has been evolved for over 5 years, and is being used by many insurance companies in Brazil. The product is being developed in JAVA and frameworks associated with Java (e.g., Spring framework⁴). The code size is approximately 0.49 million lines of code.

3.1.2. Case company 2: Diyatech

Diyatech⁵ is a medium-sized software vendor based in Pakistan. It has over 50 employees based in Pakistan. They develop performance enhancement applications for distributed environments. We studied one product of Diyatech: Product C. Product C is designed to enhance performance and support extreme transaction processing for applications running in distributed environments. It is being developed by a 6-8 member development team. It is a mature product having customers in the USA and EU in finance, health and airline industries. It is being evolved for over 10 years. Product C is developed in .Net platform⁶. Some code is also written in Java, C++ and PHP. The code size is approximately 1.5 million lines of codes.

3.1.3. Case company 3: TSoft

Case company 3, referred here as TSoft due to the company's confidentiality requirements, is an offshore development center of a transport company based in Norway. It is a new setup to develop an application for the parent company in Norway. One of the senior executives in Norway acts as the product owner, and interacts with the development center. The parent company in Norway is the main customer. However, the long term aim is to develop and market the product to outside world as well. The development team consists of a project manager, four developers, one QA engineer and a UI designer. The first release of the product was round the corner when this study was performed. The product is being developed in .Net platform. For mobile front ends, Android⁷ and iOS⁸ are used.

3.1.4. Unit of analysis

The unit of analysis is the effort estimation process used in the case organizations. Expert judgment based effort estimation is practiced in all three case organizations.

At infoway both teams use planning poker to estimate selected tasks during a monthly sprint plan meeting. This sprint plan meeting normally takes place during the first week of each month. Teams use the last week of each month for retrospectives, and for converting the backlog items to tasks that can be estimated and assigned to individual team members in the next meeting. The central two weeks are used by team members to develop and test the tasks assigned to them. The customer department interacts with the clients to collect the new requirements and feedback. The project manager is responsible for interacting with the clients, mainly through customer department, and if required directly as well.

³<https://go.java/index.html>

⁴<https://projects.spring.io/spring-framework>

⁵<http://www.diyatech.com>

⁶<https://www.microsoft.com/net/>

⁷<https://www.android.com>

⁸<https://developer.apple.com/ios/>

In parallel with the planned tasks for a sprint, Infoway teams have to work on the immediate unplanned customer requirements as well. In order to account for this continuous flow of unplanned requirements, managers set the focus factor for a team. A focus factor of 60% for a team means that 60% of the time in the sprint is allocated for the planned tasks, while the remaining 40% is reserved for the immediate customer requests. The focus factor for Infoway teams varies from 50% to 80%.

At Diyatech, the development unit consists of one development manager, architects, development and QA (Quality Assurance) teams and their respective leads. The marketing team identifies the features for the next release, and translate them to specific requirements. The development unit also gives their input in this feature discovery phase. The development manager is responsible for estimating the requirements for the next release. He consults architects and leads of the development and QA teams during the estimation process. One release consists of three to four sprints, wherein one sprint is three to four weeks long. In the last sprint of every release emphasis is on validation of the implemented functionality.

At TSoft the project manager maintains the product backlog with the help of the product owner in Norway. The team uses two weeks time boxed sprints to manage the development. At the end of each sprint, the developers select relevant items from the backlog for the next sprint, and individually prepare an effort estimate. The selected items and their estimates are discussed in a sprint plan meeting at the start of each sprint.

3.1.5. Commonalities in three cases

The following three aspects are common in the three case organizations:

- Some variant of expert judgment based approach is used to estimate effort.
- The organizations follow agile practices such as sprint and release planning, time boxed sprints, stand ups and retrospectives, unit testing, backlog etc.
- Effort is mostly underestimated, i.e. actual effort exceeds the corresponding estimate. This was highlighted upfront at the start of our collaboration in all three cases. Later, every interviewee in all three cases confirmed it. This is in line with the evidence in the effort estimation literature, which shows that underestimation occurs more frequently as compared to other scenarios (e.g., [3, 33]).

3.2. Research questions

The overarching research goal is to improve the expert judgment based effort estimation process in an agile context. We focused on expert judgment for being the frequently used estimation approach in software industry in general [2], and for the agile context in particular [8, 3]. The context is agile software development as agile practices are widely used in software industry these days.

After understanding the research context described above, the following specific research questions were framed to guide this study:

- RQ1: How to develop and evolve a customized checklist to support expert judgment based effort estimation?
- RQ2: How useful is the checklist in improving expert judgment based effort estimation?

3.3. Data collection methods

We used multiple data collection methods in all cases to ensure method and source triangulation [27]. The employed methods are listed below.

1. Interviews: We used semi-structured interviews at multiple stages in all cases: First, with a senior manager to understand the organization, the case context and an overview of the estimation process, and then later with the team members for more specific elicitation, feedback and validation purposes.
2. Workshop: We used the workshop to present and explain the estimation checklist to the team members, and also to take their initial feedback.

Table 2: Description of data collection mechanism in the case organizations

Organization	Data collection
Infoway	In the beginning two interviews over Skype were conducted with the company director to understand the company profile and portfolio, and planning and estimation processes. The study plan was also agreed upon in these discussions over Skype. Later, the first author visited the company in Brazil to conduct the face to face interviews and workshops. For product A, we interviewed two developers, team lead and the project manager. For Product B, we interviewed the team lead and three developers. The workshop was conducted jointly with representatives from both teams. The feedback on the checklist was subsequently obtained through an iterative process using questionnaire (filled by the teams jointly) and interviews with the team leads. Team leads selected the developers for interviews. The effort metrics (actual and estimated effort in hours) were collected by the respective team leads from the project management tool used in the organization.
Diyatech	To start with two interviews over Skype with the development manager were conducted focusing on the company profile and portfolio, and planning and estimation processes. Later, the first author visited Pakistan to conduct interviews and workshops with the development manager. The interviews with the architect and the developers were conducted in the written form, i.e. open ended questions were handed over to them through development manager (see interview questions in Appendix A). The development manager selected the developers for written interviews. The feedback on the checklist was subsequently obtained through an iterative process using questionnaire and interviews with the development manager. We did not interact directly with the development team in this case, as they were not involved in the estimation process. The effort metrics (actual and estimated effort in hours) were collected by the development manager in a spreadsheet from the project management tool used in the organization.
TSoft	One Skype interview with the project manager was performed to understand the case context, and their planning and estimation processes. Later, the first author visited the company to conduct interviews and workshops with the development team. We interviewed project manager, three developers and one tester. The workshop was run jointly with these team members. The feedback on the checklist was subsequently obtained through an iterative process using questionnaire and interviews with all interviewees. Being a new offshore setup, TSoft did not maintain the effort metrics of their previously completed sprints. They, however, started doing that during this study.

3. Questionnaire: We also used questionnaires to take team members' feedback on estimation checklists and also on the process used to develop these checklists. The questionnaires are provided at the end in the Appendix A.

4. Metrics: We obtained the effort related data from the archival records of the companies.

5. Checklist data: We operationalized the estimation checklist as a spreadsheet so that the usage data is stored.

The data collection methods listed above were applied differently in the three case organizations due to varying contexts, availability and confidentiality. The details are described in Table 2.

Table 3 presents the demographic information of the study participants for all three cases. Although the three case companies followed agile practices, they do not have typical agile roles (e.g., Scrum master). In case of Infoway teams, developers, team leads and the project managers are involved in sprint planning and estimation. In Diyatech case, the development manager in consultation with the architect and the team leads performs the release planning and estimation. Lastly, in TSoft case developers and project managers perform the sprint planning and estimation.

3.4. Data analysis

We analyzed the qualitative data gathered through interviews and questionnaire by applying coding [35]. We used the existing knowledge on effort estimation (e.g., known reasons for effort overruns) to manually assign interpretive and pattern codes [35] to the interview text. The data obtained in the interviews was used to describe the effort estimation process and the factors that should be considered during effort estimation process. The estimation process, and the factors organized as a checklist were validated with the teams in workshops to ensure the correctness. The qualitative data in the questionnaire was validated by conducting follow up interviews in all cases. The quantitative data collected through questionnaire about the relevance of each factor in checklist was analyzed by using simple frequency statistics.

Table 3: Demographic information of the study participants

No.	Role	Exp. in current company (Years)	com- Total exp. in industry (Years)	No. and type of interviews
<i>Infoway - Product A</i>				
1	Project manager	1.5	5	one face to face only (1 hr)
2	Team lead	3.5	3.5	several (both face to face and Skype)
3	Developer 1	2	2	one face to face only (45 mins)
4	Developer 2	4	5	one face to face only (30 mins)
<i>Infoway - Product B</i>				
1	Team lead	1	4	several (both face to face and Skype)
2	Developer 1	1	4	one face to face only (45 mins)
3	Developer 2	1	5	one face to face only (25 mins)
4	Developer 3	0.5	3.5	one face to face only (25 mins)
<i>Diyatech</i>				
1	Development manager	13	13	several (both face to face, phone and Skype)
2	Architect	10	10	written open ended
3	Developer 1	1	4	written open ended
4	Developer 2	3.5	3.5	written open ended
5	Developer 3	5	5	written open ended
6	Developer 4	3	6	written open ended
<i>TSoft</i>				
1	Project manager	1	3	several (both face to face and Skype)
2	Developer 1	0.5	2.5	one face to face (1 hr) and one Skype (20 mins)
3	Developer 2	1	3.5	one face to face (45 mins) and one Skype (15 mins)
4	Developer 3	0.5	2	one face to face (30 mins) and one Skype (15 mins)
5	Quality engineer	0.5	0.5	one face to face only (30 mins)

We used BRE (Balanced Relative Error) to calculate the accuracy of the effort estimates. Historically, the Magnitude of Relative Error (MRE) has been frequently used for estimation accuracy. However, it has recently been criticized for its biased treatment of over and under estimation [36, 37, 38]. BRE balances the over and underestimation, and have therefore been used in many recent studies on effort estimation [39, 40]. BRE (Equation 1) calculates the magnitude of the estimation error, while BREbias (Equation 2), in addition to the magnitude of error, also shows the direction (over or under) of estimation error.

$$BRE = \frac{|\text{actual effort} - \text{estimated effort}|}{\min(\text{actual effort}, \text{estimated effort})} \quad (1)$$

$$BRE_{bias} = \frac{(\text{actual effort} - \text{estimated effort})}{\min(\text{actual effort}, \text{estimated effort})} \quad (2)$$

We used BREbias in this study, as we are interested in both magnitude and direction of the estimation error.

To select an appropriate statistical test, we applied normality tests (Shapiro-Wilk and Kolmogorov-Smirnov) in SPSS⁹ on BREbias values of estimates from all cases, which indicated that the tested data are not from a normally distributed population. Therefore, we selected a non-parametric test (i.e., Mann-Whitney test [41]) to check if there is any statistical difference between the accuracy of effort estimates before and after the introduction of the checklists in the three case companies.

3.5. Characterizing the three cases with the Agile Estimation Taxonomy (AET) [32]

In Table 4 we applied the AET (briefly described earlier in Section 2.4) to characterize the effort estimation process of the three case companies included in this study. The characterization includes the details about the four taxonomy dimensions, i.e., context, effort predictors, estimation technique and effort estimate.

⁹<https://www.ibm.com/analytics/spss-statistics-software>

Table 4: Using AET to characterize estimation process of the three cases

Dimension	Facet	Infoway	Diyatech	TSoft
Context	Planning level	Sprint	Release	Sprint
	Main estimated activity	Implementation and Testing (I&T)	All	I&T
	Agile Method(s)	Customized Scrum	Customized Scrum	Scrum practices
	Product domain	eHealth	Performance enhancement	Transportation
	Project setting	Co-located	Co-located	Co-located
	Estimation unit	Task	Requirement	Task
	Number of tasks/reqs.	Refer to Table 9	Refer to Table 12	21 tasks
	Team size	5-6	6-8	7
Effort predictors	Refer to their final checklists in Appendix A to see the effort predictors used during effort estimation			
Estimation technique	Technique name	Planning poker	Expert judgment	Expert judgment
	Type	Group	Individual	Group
Effort estimate	Estimation unit	Hrs	Hrs	Hrs
	Estimate type	Point	Point	Point
	Accuracy measure	BREbias	BREbias	BREbias
	Effort data (actual, estimate and accuracy level achieved)	Refer to Table 9 and Table 12 to see the details		

4. Results and Analysis - RQ1

This question aims to describe the process that we used to develop and evolve estimation checklists in the three case organizations. The process consists of five activities (see Figure 1): understand estimation context, develop and present checklist, validate checklist statically, validate checklist dynamically, transfer and follow up. We describe these activities, how they were applied in the three case companies, their inputs and outputs in the following sections. The concepts of static and dynamic validation are adopted from Gorshek et al.'s [42] technology transfer model.

4.1. Understand estimation context

The purpose of this activity is two-fold: 1) to understand the current effort estimation process and the context in which it is carried out, 2) to elicit the factors that should be included in the checklist.

Once the checklist is developed, it needs to be implemented and integrated with the estimation process of the case company. To do that effectively, it is important to understanding the current estimation process and the context (e.g., at which stage, release or iteration, estimation is performed) in which it is carried out.

Checklist factors can be identified in different ways, such as:

- **Top down:** In top down approach, effort and cost drivers from the effort estimation literature (e.g., [4, 5]) can be used as factors to include in the checklist. Traditional cost estimation models use these drivers in arriving at the cost and effort estimates. However, in expert judgment based approaches experts use the contextual information to determine the effort estimates. The general purpose cost drivers are not relevant for all contexts.
- **Bottom up:** In bottom up approach, the factors are identified directly from the software experts involved in the estimation process. Understandably, factors identified in this way are likely to be different for each case. As the factors are elicited from the experts involved in the estimation process, they would be relevant for the given context.
- **Hybrid:** In this approach a combination of the above two approaches can be used. The factors from the literature can be used as starting point for discussion. Based on the discussion with the experts, additional factors can be identified.

Hales et al. [29], for the development of medical checklists, suggested to based the checklist contents both on the published literature and the local hospital policies and procedures. We also used a hybrid approach to elicit relevant factors for estimation checklist. The factors identified in our previous works (see Section 2 and Table 1) were used to guide the discussion in the semi-structured interviews with the study participants to identify additional factors. The results from all cases are presented in Table 5. We found that the requirements related issues are considered as the main factor behind effort overruns by all interviewees from the three case companies.

Figure 1: Checklist development process.

Table 5: checklist factors identified through interviews

No.	Factor	Infoway		Diyatech	TSoft
		Product A (n* = 4)	Product B (n = 4)	(n = 6)	(n = 5)
1	Requirements issues (changes, missing details etc.)	4**	4	6	5
2	New type of tasks involve more uncertainty	4	3	4	4
3	Overestimating the available skills/knowledge in team	3	4	3	3
4	Missing tasks while decomposing requirements	3	3	2	3
5	Ignoring non-functional requirements	3	3	2	3
6	Architecturally complex tasks introduce more delays	3	3	6	4
7	Large sized tasks are more prone to effort overruns	2	2	1	2
8	Team members working on other projects/tasks in parallel	2	1	-	2
9	New team members	2	2	2	-
10	Low team productivity during recent sprints	1	1	2	1
11	Involvement of external interfaces	-	-	-	5

*n is the number of team members who were interviewed.

**The numbers in the table denote the number of interviewees who considered a particular factor as relevant for estimation.

Infoway.

Infoway teams use planning poker, an expert judgment based group estimation technique, to estimate effort. We elicited details of the estimation process initially during Skype interviews with company director, and later during face to face interviews with team leads and members (see Section 3.1 for details).

In order to elicit the factors that should be considered during estimation, we asked study participants to discuss and describe the reasons for the most frequent estimation scenario (i.e., effort overruns) in their teams. The identified factors from the two Infoway teams are listed in Table 5. Four factors (No. 3, 4, 5 and 7 in Table 5) overlap with the previous works. The remaining factors, such as No. 1 and 2, have been described as the leading reasons for effort overruns in other studies as well (e.g., [3]).

Diyatech.

At Diyatech the development manager uses expert judgment to estimate requirements for the next release. He takes input from the architect and the team leads. The developers are not involved in the estimation process (see Section 3.1 for more details). The development manager and the architect have been working in the same company for over ten years. The factors that contribute to the inaccurate effort estimation are described in Table 5. These were used as input to the first draft of the estimation checklist for Diyatech.

TSoft.

At Tsoft, team members prepare estimates for the selected backlog items for the next release. The estimates are discussed and revised, if required, in a sprint meeting. In the opinion of the development team, the factors that contribute to the effort overruns are listed in Table 5. These factors overlap with the factors elicited from the other two cases. The one additional factor is related to the external interfaces in case of TSoft. The system being developed had to interface with several third party systems, and the tasks that involve these interfaces were more complex. We used these factors as input to develop the first version of the estimation checklist for TSoft.

4.2. Develop and present initial checklist

The purpose of this activity is to use the output of the previous elicitation step to formulate and present a customized checklist for the involved teams. The checklist would include factors that should be considered by the team when estimating requirements. It is an iterative step wherein the proposed checklist should be revised based on teams' feedback. In case there are no more change suggestions, the involved teams are requested to prepare for the next step, i.e., static validation, wherein they will have to apply the checklist to estimate a sample of old tasks or requirements.

While presenting the checklist to the involved teams, it should also be discussed how exactly the checklist will be used by the teams during effort estimation.

Infoway.

The first version of the Infoway estimation checklist is described in Table 6. The last column in Table 6 links checklist factors with the factors we identified through interviews. Since it was enough to show this linkage once, we did not include this last column in other checklist tables to keep them short.

We conducted a joint workshop with both teams to present the checklist and take participants' initial feedback. We presented each checklist item by focusing on three questions: What (is the factor), Why (is it there in the checklist), and How (is it to be answered during estimation). During the discussion on the "how" part, we discussed suitable scale values that teams would be required to select for each factor while filling in the checklist to estimate a task or requirement. It was agreed that the checklist would be operationalized as a spreadsheet. The team leads would use its factors to guide estimation discussions and would also answer each factor with a consensus value.

The participants agreed on the following changes (see rows in gray background in Table 6) during the workshop:

- Remove Factor 10, and add it as last scale value in place of "Lacking" with Factor 8. We therefore removed Factor 10 and redefined the scale of Factor 8 as: Very good, good, average, below average and need to talk to customer before moving on.
- Add a new factor to identify the type of the task, i.e., implementation or testing.

We incorporated these recommended changes in the checklist, and handed it over to the teams for static validation (see next Section).

In the light of the discussions during workshop, we also added detailed notes in the spreadsheet with each checklist factor explaining the factor and corresponding scale values and when a value should be selected.

An example note with the checklist factor about domain knowledge:

It is the information/knowledge about the domain of the product that you are developing. For example in case of ehealth products, information on the health system, insurance policies, rules etc. represent domain knowledge

Select "Very good" if you think team has all the domain knowledge required to implement this task
Select "Good" if you think team has most of the domain knowledge required to implement/test this task
Select "Average" if you think team has to study/research for some time to obtain the required domain knowledge
Select "Below Average" If you think team has to study/research more to obtain the required domain knowledge
Select "Lacking" if you think team needs to study/research a lot or need help from seniors or other sources to be able to get the required domain knowledge.

Diyatech.

The first version of the checklist is described in Table 7. It was shared with the development manager for initial feedback. Later, it was also discussed in one workshop session with the development manager wherein the first author presented each checklist item and its corresponding scale values. Besides describing each checklist factor and the reason behind its inclusion in the checklist, we also explained how can it be answered. Using the knowledge and experience from the Infoway case, we also described and presented the explanations for how to answer each checklist factor by selecting the appropriate scale values. Checklist was implemented as a spreadsheet in this case as well. It was discussed that the development manager would first answer all checklist factors with suitable values before proposing the most likely effort estimate for each requirement.

The following changes were suggested during this workshop:

- Remove Factor 10 as it is not relevant to the type of system being developed in the case under study.
- Remove Factor 13 as the team members do not work on other projects in their context.

We incorporated these changes in the checklist and handed over the revised version (Table A.3 in Appendix A) to the development manager for static validation.

Table 6: Infoway checklist V1

No.	Checklist factor	Scale	Linked factor no. in Table 5
1	How similar is the task to the previously developed tasks?	To a great extent, somewhat, very little, not at all	No. 2
2	Rank team's technical competence/skills required to implement and test this task	Very good, good, average, below average, lacking	No. 3
3	Rank team's domain knowledge required to implement this task	Very good, good, average, below average, lacking	No. 3
4	Does the task require an architectural change?	Yes, not sure, no	No. 6
5	Does the task involve communication between multiple (sub) systems?	Yes, not sure, no	No. 6
6	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no	No. 6
7	Are there stringent non-functional requirements (performance, security etc.) associated with this task?	Yes, not sure, no	No. 5
8	How clear is your understanding of the task?	Very good, good, average, below average, lacking	No. 1
9	Is the task size suitable to fit in the planned sprint?	Yes, not sure, no	No. 7
10	Is there a need to communicate with customer to further clarify this task?	Yes, not sure, no	No. 1
11	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no	No. 8
12	Does the team has new member(s)?	Yes, no	No. 9
13	Rank team's recent productivity	Very good, good, average, below average, lacking	No. 10
Based on the above characterizations, most likely effort estimate for this task is: _____ hours			

Table 7: Diyatech checklist V1

No.	Checklist factor	Scale
1	The requirement is classified as:	x, y, z (company specific categories)
2	How similar is the requirement to the previously developed requirements?	To a great extent, somewhat, very little, not at all
3	Rank team's technical competence required to implement this requirement	Very good, good, average, below average, lacking
4	Rank team's technical competence required to test this requirement	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this requirement	Very good, good, average, below average, lacking
6	Rank team's recent productivity	Very good, good, average, below average, lacking
7	Does the team has new member(s)?	Yes, no
8	Will this requirement's implementation involve an architectural change?	Yes, not sure, no
9	Will this requirement's implementation involve communication between multiple (sub) systems?	Yes, not sure, no
10	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
11	The requirement is classified as Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	Only an NFR, FR with major NFR constraints, FR with minor NFR constraints, only FR
12	How clear is your understanding of the requirement?	Very good, good, average, below average, need to talk to customer before moving on
13	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no
14	Is the requirement size suitable to fit in the planned release?	Yes, not sure, no
Based on the above characterizations, specify your most likely effort estimate for the following activities:		
	Requirements analysis	_____ hrs
	Documentation	_____ hrs
	Design	_____ hrs
	Coding	_____ hrs
	Testing	_____ hrs
	Non-development work	_____ hrs
Total		_____ hrs

Table 8: Tsoft checklist V1

No.	Checklist factor	Scale
1	What is the type of the task?	UI, Business logic, DB
2	How similar is the task to the previously developed tasks?	To a great extent, somewhat, very little, not at all
3	Rank team's technical competence/skills required to implement and test this task	Very good, good, average, below average, lacking
4	Rank team's domain knowledge required to implement this task	Very good, good, average, below average, lacking
5	Will the task implementation likely to involve interfacing with external services/systems?	Yes, not sure, no
6	Does the task require an architectural change?	Yes, not sure, no
7	Does the task involve communication between multiple (sub) systems?	Yes, not sure, no
8	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
9	Are there stringent non-functional requirements (performance, security etc.) associated with this task?	Yes, not sure, no
10	How clear is your understanding of the task?	Very good, good, average, below average, lacking
11	Is the task size suitable to fit in the planned sprint?	Yes, not sure, no
12	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no
13	Rank team's recent productivity	Very good, good, average, below average, lacking
Based on the above characterizations, most likely effort estimate for this task is: _____ hours		

TSoft.

The first version of TSoft checklist is described in Table 8. It was presented to the development team in a workshop held in the company. We used the same approach as discussed above in the Infoway case to present checklist to the TSoft team. TSoft team members individually prepare initial estimates of their tasks before discussing them in joint meeting. It was agreed that the team members would first use the checklist while preparing their individual estimates, and later to present and discuss their estimates in the joint meeting.

The following changes were proposed in the workshop:

- Revise Factor 1 on task classification to include more options in the available scale points. Previously, we had just frontend, backend and business logic as task types. The team members suggested to include a combination of these types as additional option.
- Remove Factor 13 about team recent productivity. The team members opined that the recent productivity could be bad due to issues outside their control, such as ambiguity in requirements, and therefore should not be a factor in determining the estimates.

We incorporated these recommendations in the checklist, and handed over the revised checklist (Table A.5 in Appendix A) to them for static validation.

4.3. Validate checklist statically

In static validation the involved team(s) is asked to use the checklist to estimate a sample of tasks or requirements from previous sprints. The idea is to allow the teams to use the checklist in a close to real situation to develop a good understanding of how to use the checklist to estimate tasks. After the validation, team's feedback is elicited focusing on the suggestions for further improvement. In case of major changes, the revised checklist should be tried again by the involved teams. At the end of this step, the teams should be requested to prepare for using the revised checklist in real context.

Infoway.

At the end of the workshop described above in Section 4.2, both teams were requested to select a sample of at least three tasks from previous sprints, and estimate them with the checklist. It was decided that the two team leads would perform this task, and would act as the contact persons for this study. They would themselves interact with the teams, if and when required.

We held a discussion with both team leads where they were requested to explain how they used the checklist, and provide their feedback. The team leads expressed positive opinion on the checklist, and both shared their desire to use the checklist dynamically during the next sprint plan meeting. Furthermore, changes in the two factors were suggested and agreed.

- The first change is about Factor 7, which deals with non-functional requirements. Team leads suggested that it is possible that a task is entirely about a non-functional requirement or functional requirement or both. Further, this factor should also be moved to the top of the list to identify the task type upfront.
- The second change is about the newly added factor during the workshop, which is about type of the task. The team leads explained that sometimes a task is assigned to a team member to do some research on it. To incorporate that, we added a new scale value “Research” besides implementation and testing in this factor.

The revised checklist is displayed in Appendix A (Table A.1) with revised factors marked with the gray background. The team leads expressed their teams’ readiness in using the revised checklist during the next sprint plan meeting.

Diyatech.

After using the checklist to estimate a sample of requirements from a previous release, one more factor was suggested to be included in the checklist. The factor is about priority of the requirements with respect to the importance of the customers to the product and company. The requirements from the old and large customers are treated as high priority items. The development manager opined that this should be reflected in the checklist. We added this factor in the revised checklist (Table A.3 in Appendix A). The development manager expressed his interest to continue to use the checklist in actual planning.

TSoft.

All involved members at TSoft used the checklist to estimate a sample of three previous tasks. After this exercise, we held a face to face session with them. The only change requested by two members of the team was to re-arrange the checklist factors by moving up the factor about task understandability. We re-organized the checklist according to the sequence suggested by them. The team members expressed confidence and interest in using the checklist during the next iteration planning instance.

4.4. Validate checklist dynamically

The purpose of dynamic validation is to start using the estimation checklist in real context, i.e., during estimation and planning meetings to estimate tasks or requirements. After teams have practically used the checklist with their effort estimation process for some time, one can analyze its usefulness and obtain teams’ feedback on their interest using and evolving it in future. It is an iterative step wherein teams can continue to evolve the checklist in light of their experiences to add or remove or modify checklist factors. Checklist should not be perceived as a final or static document. It is important to request the involved teams to keep record of the effort data, so that after the release or iteration is complete, we are able to analyze the estimation accuracy.

Infoway.

The revised checklist, described in Appendix A (Table A.1), was handed over to the team leads as two separate spreadsheets. It was discussed and agreed that the team leads would ensure that each checklist factor is characterized before estimating a task during estimation meetings. The first author stayed with the teams in Brazil until this stage.

After both teams had used the checklist with two sprints, we contacted team leads to take their feedback. The checklist’ perceived benefits and its impact on accuracy are discussed in Section 5. Both teams expressed keen interest in continuing to use the checklist in future. The following changes were suggested by the two teams.

- The teams suggested to add another factor to deal with the issue of legacy code. Since both teams are working on relatively old products, a considerable amount of legacy code has been developed. The tasks that involve changing or understanding legacy code are more effort intensive. We therefore added a new factor in the checklist about legacy code.
- The teams opined in cases where there will be relatively less time available for estimation meetings, the use of checklist adds some time overhead. To deal with such scenario it was discussed to classify the checklist factors into two parts: mandatory and optional. Mandatory part consists of those factors that will be characterized in all situations, while optional part includes factors that will be filled subject to the availability of the time. Furthermore, the team leads opted to do this classifications by themselves after consultations with their respective teams. The two leads finalized a common checklist consisting of the two parts, mandatory and optional.

The checklist was further revised (see Table A.2 in Appendix A) in the light of the changes mentioned above, and handed over to the Infoway teams for future use.

Diyatech.

The revised checklist was used in estimating requirements of one release. The development manager and the architect suggested the following two changes in the checklist:

- Some requirements involve designing and implementing complex distributed algorithms. These requirements are found to be more challenging. They suggested to add a new factor to cover such cases.
- Another factor, related to the interruptions caused by meetings and other company events, was highlighted. These interruptions affect the momentum of the team members due to the switching of the context from the work.

We added the two suggested factors in the checklist, and handed over the revised version of the checklist (Table A.4 in Appendix A) to the development manager. The encouraging sign for us was their interest in continuing to use the checklist in future as well.

TSoft.

TSoft team used the estimation checklist in two sprints to estimate tasks. The team did not suggest any change in the checklist. They expressed their interest in using the checklist with future sprints as well.

4.5. Transfer and follow up

After the estimation checklist has been dynamically validated, the next step is to revise and transfer the finalized checklist to the teams. Furthermore, it is important to submit a study report to the company management describing the whole exercise and its outputs in the form of final checklist and its version history. The report should also explain the benefits of using estimation checklists based on the analysis of the data collected during the whole process. It is up to the company management to see if they are interested in officially adding the checklist instrument as part of their estimation process.

Besides transferring the checklist to the involved teams, it is also important to transfer the process used to develop the estimation checklists. If the teams understand all activities of the process and their purpose, they can themselves apply the process to develop or evolve estimation checklists for different contexts (e.g., new project).

A further follow up step can be performed after some time in future. The aim is to determine if the checklist is still in use or being evolved, and how useful it has been in improving the effort estimation process.

463 *Infoway.*

464 Given that the teams have expressed their interest in continuing the use of estimation checklist after dynamic
465 validation step, the finalized checklist is transferred to the teams for their use. Furthermore, study report was submitted
466 to the company management that includes the finalized checklist, version history, its benefits based on the analysis of
467 the qualitative and quantitative data (see Section 5) collected at different stages of this study.

468 We did a follow up after a gap of six months. The team leads informed us that the checklist has become a part
469 of their estimation and planning process. It is, however, being used in a different form. Both teams have printed the
470 checklist on a single page that each team member uses during the estimation. Further, the checklist had also been
471 translated to Portuguese language to facilitate some team members. During the follow up interaction, the Product A
472 team lead shared his experience that checklist helped him to get back into the estimation process when he came back
473 from vacations.

474 *"I've spent some time not participating in estimation meetings and then I went back for it. At that moment, the
checklist really helped me, because I had spent some time far from the estimation meeting and then I had to do it
again, it was much easier to adapt myself back again."*

Product A team lead

475 After transferring the final checklist and the follow up, we also requested both team leads to provide their feed-
476 back on the checklist development process. First we emailed them a document (see data collection instrument C in
477 Appendix A) briefly describing the purpose of the feedback, the checklist development process and a list of ques-
478 tions. The team leads were requested to discuss the questions within their teams and provide their reflections. After
479 receiving back their reflections, we conducted semi-structured interviews with team leads to confirm and discuss their
480 feedback. In nutshell, both team leads expressed their confidence that they not only understand the process, but can
481 also apply it by themselves to develop or evolve estimation checklist for a different project. During the interviews we
482 ensured that the team leads understand each step, its purpose and how can it be applied.

483 Product A team lead suggested that the step 3 (static validation) of the process should be iterative, i.e., we should
484 only proceed to the next step when the team is satisfied about the checklist and have developed a shared understanding
485 of the checklist. This was our intention as well when we described in step 3 of the process that, based on team's
486 feedback, the checklist should be revised and shared with the team. We proceed to the next step only when the team
487 is satisfied with the revised checklist. Product B team lead suggested that for future applications of this process, the
488 revised checklist and its previous versions should be made available at a shared space (e.g., company intranet). The
489 team members shall be able to read comments of other members and also have the option to provide their feedback.
490 This will allow the team to preserve the revision history of all checklist versions.

491 *Diyatech.*

492 After the dynamic validation step, we handed over the revised checklist to the development manager, who shared
493 with us that the checklist would remain part of their estimation process. We highlighted the importance of maintaining
494 the estimation data over the next few releases for a potential follow up investigation.

495 We did a follow up after a gap of about six months. The development manager informed us that they have recently
496 used the checklist again on a recent release of Product C. During this follow up we also requested the development
497 manager to provide us feedback on the process we used to develop and evolve the estimation checklist. Like the
498 Infoway case, we first send the feedback document (see data collection instrument C in Appendix A) and then also
499 conducted a Skype interview. The development manager shared with us that due to his close involvement in this study
500 and straightforwardness of the process, he fully understands the process activities and is confident that he can himself
501 use it to develop estimation checklist for a different project, if required. The development manager highlighted the
502 need to understand the project in depth for which the checklist is to be developed. During the follow up interview, the
503 development manager stressed the importance of carefully identifying the relevant checklist factors, as part of step 1
504 of the process, with the help of the team members who understand project complexities and context.

505 *"Yes, the activities mentioned pretty much cover the steps required to create a checklist that works for the most of
the projects. However, since every project can have its own nature of complexities, an in-depth understanding of
that project may help develop the "variable" part of checklist for that specific project."*

Development Manager

506 *TSoft.*

Figure 2: Checklist versions.

The revised checklist was handed over to the project manager and the development team. We also provided the project manager with a report of the study findings using estimation data after the use of checklist and developers' subject opinion. The report also included suggestions relating to the planning process, code review process, and the need to report estimates and actual effort for monitoring performance and using the data to improve planning and estimation in future. The TSoft team was not using any project management tool to record estimates and actual effort spent, we requested them to enter the estimates, and subsequently the actual effort in the spreadsheet, which included the checklist as well.

After a gap of six months, we contacted the project manager of the TSoft team through email for follow up. In a half hour Skype interview subsequently, we were informed that the development team is not using the checklist to estimate the tasks during sprint planning. According to the project manager the developers took it as an overhead to: 1) use the checklist to estimate, 2) and more importantly, to log the actual hours spent on each task. Keeping track of the actual effort spent was a requirement from our side, as we wanted to determine the accuracy of the effort estimates.

Nonetheless, the project manager shared with us that he himself is using the checklist factors in prioritizing the product backlog. The project manager uses the checklist factors to perform an initial impact analysis of the backlog items to determine their priority. Some checklist factors such as architectural change, involvement of NFRs and external services, support in establishing an initial understanding of the complexity of the involved tasks. Other factors, such as team's skills and domain knowledge, helps in ranking the team's readiness and capacity to take on the particular backlog item.

We reformulated the checklist factors by referring them to backlog items (see Table A.6 in Appendix A), rather than sprint tasks, so that they are aligned with the revised usage of the checklist.

We contacted the project manager again after a gap of about six more months to take his feedback on the checklist development process. Like the previous two cases first, we sent him the feedback document (see data collection instrument C in Appendix A). We also performed a Skype interview with the project manager. As mentioned previously, the checklist is not being used by the development team during effort estimation. The focus of the discussion was on ensuring that the project manager understands the process well enough so that he can himself use it in future if required (e.g., for a new project). The project manager did not suggest any change in the checklist development process and observed that due to his involvement in the study, he has a good understanding of the process. Two aspects of the process that he particularly appreciated are its focus on understanding the project context directly from the team members and pilot testing during static validation before actually using the checklist.

5. Results and Analysis - RQ2

This section presents and analyzes the results for RQ2, which is about usefulness of the checklist in improving the expert judgment based effort estimation in the three case organizations. In all three cases the study participants showed keen interest to continue the use of estimation checklist in real situations (dynamic validation) after the workshops and the static validation. The checklist was then used by the study participants in real situations. Based on this real use, we collected more data to see how useful has been the checklist in improving the effort estimation.

In all three cases we first used a questionnaire, which included questions about relevance of each checklist factor to the effort estimation process, and also the feedback and reflections on the checklist as a whole (see feedback instrument in Appendix A). Further, we also collected effort data to see if the use of the checklist has made any impact on the accuracy of the effort estimates. Finally, we performed Skype interviews to further understand the results and also the observed benefits and drawbacks, if any, of the checklists. We summarized the benefits noted by the case participants in Table 10.

The inclusion of the recommended changes in the different versions of the checklists has been discussed earlier in Section 4.

5.1. Infoway

After the revised estimation checklist (Table A.2 in Appendix A) was used by both Infoway teams in three sprints, we collected data to see how useful was the checklist.

Table 9: Accuracy of estimates before and after checklist introduction at Infoway

Sprint	Product A			Product B		
	No. of tasks	Mean BREbias	Median BREbias	No. of tasks	Mean BREbias	Median BREbias
<i>Sprints before the use of checklist</i>						
Sprint 1	25	0.80	0.81	28	1.13	- 0.06
Sprint 2	21	0.62	0.57	25	- 0.36	- 0.09
Sprint 3	33	0.25	0.09	32	2.11	1.06
Sprint 4	26	- 0.04	0.11	63	1.30	0.15
Sprint 5	48	0.25	0.14	34	3.21	1.20
<i>Sprints where checklist was used in estimation</i>						
Sprint 1	40	- 0.11	0.05	88	0.07	- 0.02
Sprint 2	26	- 0.26	- 0.13	40	0.29	- 0.01
Sprint 3	28	- 0.02	-0.14	86	0.35	0.12

5.1.1. Checklist usefulness based on subjective feedback:

A common concern of both teams on the initial two versions of the checklist was time overhead due to the use of checklist during estimation. This was addressed by revising the checklist to include mandatory and optional parts.

"It takes more time to finish the estimation process. Less items to consider would be better. "

Product A team lead

The teams, after using the revised checklist for two sprints, found that the checklist provided many benefits. They noted that the checklist, besides improving the estimation process and their confidence in estimates, has also helped them in improving the understanding of the tasks being estimated. The team leads associated it with the fact that checklist induces discussions on various topics (e.g., architecture, legacy code) covered in various factors.

"Checklist has an impact improving confidence about estimates, since it provides an additional basis for the estimate besides each developer's experience. "

"It helps understanding the task, it helps remembering points that everybody knows, but not always remember to consider. Besides, it increases the discussions around each task."

Product A team lead

"Checklist helped us to remember all tasks. It also improve[d] our understanding."

Product B team lead

The team leads first rated each checklist factor after consulting their respective teams. Both leads discussed their ratings with each other and shared with us their common rating. They rated factors 1 to 8 in Table 11 as extremely relevant, while factors 9 to 12 were rated as relevant to the effort estimation. They were undecided about factor 13 and 14. The remaining factors in Table 11 (15 to 17), marked as "N/A" are not rated as they were not part of the checklist for the Infoway case.

5.1.2. Checklist usefulness based on effort data:

Table 9 presents the mean and median BREbias values for sprints before and after the checklist was introduced in the two Infoway teams. The effort estimates vary from 1 hour to 8 hours normally. The company provided us with the estimation metrics for five sprints before the introduction of the estimation checklist. Apart from few exceptions, the data shows that the underestimation was the dominant trend for both products. It is much more significant in case of Product B. We have the data for only three sprints when the checklist was subsequently used to estimate tasks. We can see reduction in the underestimation bias in both products. In case of product A the data shows a shift towards the overestimation. The accuracy of the estimates in the three sprints where checklist was used, is better than the previous five sprints. However, Man-Whitney test, for both products, found that the differences between the accuracy of the two estimates (i.e., estimates with using checklist and estimates without using checklist) are not statistically significant.

Table 10: Benefits of using estimation checklist

No.	Benefit	Infoway	Diyatech	TSoft
1	The checklist improved our confidence in estimates	✓	✓	✓
2	The checklist reduced the chances of missing important factors during estimation	✓	✓	✓
3	The checklist improved our estimation process	✓	✓	✓
4	The checklist reduced the chances of missing tasks	✓	✓	✓
5	The checklist improved our understanding of the tasks being estimated	✓	-	✓
6	Expert estimation became relatively objective due to the checklist	-	✓	-
7	The checklist helped in understanding the implementation complexity of tasks	-	-	✓
8	The checklist is a good support for new team members	✓	-	-

Table 11: Teams' feedback on the relevance of the checklist factors to the effort estimation

No.	Checklist factor	Infoway	Diyatech	TSoft
1	What is the type of the task*?	++	++	++
2	The task is implementing a Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	++	++	+
3	Rank team's technical competence/skills required to implement and test this task	++	++	++
4	How similar is the task to the previously developed tasks?	++	+	++
5	Rank team's domain knowledge required to implement this task	++	+	+
6	How clear is your understanding of the task?	++	+	++
7	The implementation of the task requires understanding and/or changing legacy code	++	N/A	N/A
8	Does the task involve communication between multiple (sub) systems?	++	++	++
9	Does the task require an architectural change?	+	++	+
10	Does the team has new member(s)?	+	++	N/A
11	Will the team or some member(s) be working on other products/projects/tasks in parallel?	+	N/A	++
12	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	+	N/A	+
13	Is the task size suitable to fit in the planned sprint?	~	+	+
14	Rank team's recent productivity	~	N/A	N/A
15	The task involves designing and/or revising and/or implementing distributed algorithm(s)	N/A	++	N/A
16	The may be interruptions during the planned release/sprint affecting team member's momentum due to context switch	N/A	++	N/A
17	What is priority of the requirement being implement	N/A	++	N/A
18	The task involves interfacing with external services/systems?	N/A	N/A	++

* In case of Diyatech the word requirement was used instead of task, as they estimate requirements not tasks.

++ = strongly agree, + = agree, ~ = undecided, - = disagree, - - = strong disagree, N/A = Not Applicable

5.2. Diyatech

The revised checklist was used in one release to estimate requirements. The release consisted of 8 requirements, wherein the estimates varied from 200 hours to 800 hours.

5.2.1. Checklist usefulness based on subjective feedback:

After the estimation was performed, we collected feedback from the development manager about the usefulness of the estimation checklist. Development manager shared several benefits of using checklist during estimation (see Table 10). In addition to the benefits highlighted by Infoway and TSoft cases, development manager opined that the expert estimation became more objective due to the use of checklist, without introducing much overhead.

"The best part is that by following this checklist, the estimation becomes more factual and objective. One might skip important factors if checklist is not followed and hence becomes underestimation."

"Although, some of the checklist items were always considered but there were a few important items that we mostly overlooked...it has helped us in structuring estimation process and has moved us a step closer to be more accurate in time estimation."

Development Manager

The development manager rated the relevance of the 13 factors in the final checklist to the estimation. He rated 9 factors as extremely relevant, while the remaining 4 as relevant. These ratings (see Table 11) were obtained on the revised checklist, and therefore understandably no factors were rated as irrelevant.

Table 12: Accuracy of estimates before and after checklist introduction at Diyatech

Release	No. of requirements	Mean BREbias	Median BREbias
One release where checklist was not used	8	0.20	0.27
One release where checklist was used	7	0.02	0.04

5.2.2. Checklist usefulness based on effort data:

We managed to get the effort estimation data for one previous release when the checklist was not introduced. The mean and median BREbias values for the two releases (see Table 12) show that the checklist estimates are more accurate and have considerably low underestimation bias. The data clearly is not enough to draw any conclusions. However, it is aligned with the opinion of the manager and team that the underestimation had been the dominant trend, and checklist has helped in improving the estimates. We applied Man-Whitney test and found statistically significant difference (P value 0.026) between the accuracy of the two estimates (i.e., estimates with using checklist and estimates without using checklist). However, since the sample size in Diyatech case is quite small, the results should be interpreted with care.

5.3. TSoft

The TSoft team used the checklist for two sprints to estimate 21 tasks in total. The task estimates vary from 4 hours (half day work) to 40 hours (one week work). At the time of study, TSoft did not have a process in place to log the actual effort spent on each task. It was agreed during this study that the team members will log their time in the same spreadsheet that included the estimation checklist. Therefore, we can not compare the accuracy of checklist estimates with the accuracy of the previous estimates.

5.3.1. Checklist usefulness based on subjective feedback:

TSoft team members shared several benefits of using checklist to estimate tasks (see Table 10). Besides the common benefits that were also shared by the Infoway and Diyatech cases, TSoft team opined that the use of checklist resulted in useful discussions on the technical and architectural complexity of the tasks. The development team rated the 13 factors (see Table 11) in the TSoft revised checklist for their relevance to the estimation. They rated seven factors are extremely relevant, while the remaining six as relevant.

5.3.2. Checklist usefulness based on effort data:

The mean and median BREbias for the 21 tasks estimated with the checklist is -0.04 and -0.05. The accuracy values indicate the presence of minor overoptimism bias. The developers, during their interviews earlier, opined that previously the most common scenario had been the presence of underestimation bias. The use of checklist may have lead to the decrease in the over optimism in their estimates. However, due to the unavailability of the previous data and also limited new data (only 21 tasks) we can not conclude anything in this case.

"We get more understanding of the task type by going through these factors. The task effort is estimated more accurately by considering these factors."

Developer 1

"I think checklist help to understand the task basically wherever similar task show on sprint I get an idea how much time I needed to complete this task."

"...this checklist improve my confidence while estimating the task."

Developer 2

"Improvement in estimation and also helpful to understand the complexity of task and complete a task on time. "

Developer 3

6. Discussion

In this section first, we further discuss the results of the two research questions. Later, we describe the implications of this work for both SE researchers and practitioners.

6.1. RQ1: Checklist development and evolution

We followed a multi-step process to develop and evolve estimation checklists for the three cases included in this work. The process starts with the **understand estimation context** activity, wherein we elicited the details of the estimation process followed, and the factors that are considered during estimation through semi-structured interviews with case participants. The results (see Table 5) show that there are a number of reasons for effort overruns that are common across three cases. These common reasons include lack of details in requirements, changing requirements, overestimating team's technical competence and domain knowledge, missing tasks, ignoring non-functional requirements, architectural complexity of tasks and task size. These reasons for effort overruns have been identified in other studies as well (cf. [3, 43, 44]). There were, however, some reasons which were not common in all cases, such as involvement of external interfaces.

The next activity is the **develop and present checklist**, wherein the first draft of the estimation checklist was presented in each case to the involved teams in a workshop. The aim is to present the checklist, and describe how it can be used and more importantly take initial feedback to revise it. In this phase, participants in each case suggested changes (see Figure 2) to customize checklists to their respective contexts.

These changes should be embraced and welcomed, as it indicates the study participants' interest in the proposed checklist. At this stage, the participants have not yet used the checklist during effort estimation. It is likely that when they actually use checklist, they want to further customize the checklist. The next activity, **validate checklist statically**, provide this opportunity to the case participants to use the estimation checklist on a sample of previous tasks and provide feedback for further changes. In all three cases, the participants used the checklist on a small sample of tasks and recommended further changes. For example, in Diyatech case the development manager suggested to include a factor about requirement's priority in the checklist, and in case of TSoft the team suggested to remove a factor about team's recent productivity. This type of validation in a controlled setting has been suggested by Hales et al. [29] as well in the development of medical checklists.

During the next activity, **validate checklist dynamically**, the teams use the checklist in real context to estimate tasks/requirements for the next release/sprint. When the teams use the checklist in real context, they may feel the need to further fine tune the checklists. After the dynamic validation phase, the teams were requested to provide their feedback on the estimation checklists through a feedback instrument (see checklist feedback instrument in Appendix A) consisting of both open and close ended questions. The participants from two cases (Infoway and Diyatech) suggested further changes (see Figure 2), while TSoft did not ask for more changes. These changes further customized the checklist to the context of the respective teams and companies.

One of the concerns of the Infoway teams was about time overhead due to the use of checklist during estimation sessions. In some cases, the teams have to estimate quickly due to the limited time available for estimation. Hales et al. [29] suggested to avoid having long checklists, while Jørgensen [31] recommended that the checklist should only consist of relevant content. In a follow up discussion with the Infoway team leads, we came up with the idea to classify the checklist factors into two parts: mandatory and optional. The mandatory part consists of that minimum set of factors that are to be considered in all situations, while the optional part consist of factors that will be considered subject to the availability of the time. Infoway products are relatively old products, and considerable amount of legacy code has been developed. To account for that, the Infoway teams suggested to include an additional factor about legacy code in the mandatory part of the checklist.

Diyatech development manager suggested to add two more factors in their checklist after using it to estimate requirements of the next release. The first one is about the involvement of complex distributed algorithms, which is a common scenario in the performance enhancement product domain. The second added factor deals with the work interruptions due to company and team events, impacting the team momentum.

The last phase of the process is **Transfer and follow up**, wherein the revised checklist is handed over to the involved team for future use. It is up to the company management to decide if they want to include the checklist in their estimation process. In each case after dynamic validation, we handed over the revised checklists to our contacts in the three companies. At that stage, all had expressed their interest to keep using the checklist in future as well. As part of the follow up, we contacted the Infoway team leads, Diyatech development manager and TSoft project manager. Infoway team leads informed us that the checklist has become a part of their estimation process. It is being used by the two teams during estimation sessions. The checklist has also been translated to Portuguese to support those team members who feel more comfortable with their native language. Diyatech manager informed us that they

have used the checklist again in their recent release of Product C, and their plan is to continue using the checklist in future as well. TSoft project manager informed us that the team does not use the checklist anymore during estimation. However, he himself has been using checklist factors in prioritizing product backlog and sprint planning. The checklist factors support him in performing impact analysis of the backlog items. Dynamic validation phase of the Diyatech case was concluded relatively recently. Therefore, no follow up has yet been performed in this case.

Finally as part of the follow up, we also requested companies to provide their feedback on the checklist development process. The idea was to ensure that, besides the checklists, the proposed process for checklist development is also transferred to the companies. In all cases, the involved practitioners expressed their confidence that, if required, they can apply the process by themselves in developing new checklists.

6.2. RQ2: Usefulness of the checklist in improving effort estimation

In all three cases, the participants provided a highly positive feedback after using the checklist in real context. They noted several benefits (see Table 10) of using checklist during effort estimation. Checklist factors were found to be useful in reminding teams of the important factors that should be considered during effort estimation. The Checklist introduced some structure in the otherwise largely ad-hoc expert estimation process that was being practiced in all three cases. One of the reasons for effort overruns, shared by participants from all three cases, was forgetting tasks during the task definition/decomposition process. The participants observed that the checklist factors such as those related to non-functional requirements, testing and tasks classification into company specific types, has reduced the chances of missing tasks. These benefits of estimation checklists have been reported earlier as well [2]. Checklists have been found to be beneficial in avoiding human error in omitting relevant activities and adherence to best practices in other disciplines as well, such as aviation, product manufacturing and health care [13].

The study participants noted that the explicit consideration of checklist factors during estimation has increased their confidence in effort estimates. Passing and Shepperd [12] also noted that the use of checklist improves estimators' confidence in effort estimates.

Another benefit of using checklist noted by study participants is about improved understandability of the task or requirement being estimated. The participants believe that the checklist factors induced discussion on different topics, such as impact on architecture, involvement of legacy code or non-functional requirements and required skills, which resulted in much better understanding of the requirements. These factors were not new for the team members. However, they were discussed occasionally when someone from the team is able to recall particular factors during estimation. The inclusion of these factors in the checklist made their use more consistent.

The use of checklist has reduced the underestimation bias in two cases (Infoway and Diyatech), and resulted in more accurate estimates. The manager and team leads from these two cases attributed this improvement to the use of the checklist, which forces them to consider these factors during estimation. Previously, these factors were not considered consistently. Furulund and Moløkken-Østvold [11] also observed that the checklist improved the estimation accuracy. In case of TSoft, we do not have the previous data to perform this comparison. However, the available effort data for just 21 tasks, that were estimated using checklist, show that the estimates are quite accurate and have a minor overestimation bias.

6.3. Implications for research

During the course of these three case studies, we had the opportunity to closely work with the involved teams to develop and evolve estimation checklists to improve their expert estimation processes. We learned the following lessons during this multi-case study, which can be used by other researchers investigating similar topics:

- Provide multiple opportunities for practitioners' feedback: We used a process that allowed case participants to provide their feedback on checklist at multiple stages (e.g., during first workshop, interviews after static and dynamic validation). Garousi et al. [45] also recommended to have effective communication with the industrial partner, and to run regular workshops and seminars. The initial feedback is not enough, as it just consist of participants' first impressions on the relevance of checklist factors. When participants have used the checklist during effort estimation for some time, they can provide more useful suggestions for improvement. Therefore, it is important to try to have them use the checklist, preferably in real context, in order to get effective feedback. The on-site presence of the first author in two cases proved very helpful in establishing trust and efficiently working together to develop and try initial checklist versions.

- Incorporate suggested changes: One checklist can not fit all contexts. The context of the company and the teams differ with respect to the domain, product complexity and history, team member's experience and skill etc. Therefore, it is important that the checklist is customized for the team context, and this customization should be based on the team's feedback. At each stage, we made an effort to act on the participants' feedback (e.g., add/remove a factor from checklist, improve scales or description) and reported back with the revised checklist.
- Agile context: Most development teams use a combination of agile practices to develop software. In agile software development context the focus is on using light-weight processes and tools. Therefore if the checklist has to be used by an agile team as support to expert estimation, it should not consist of long list of factors. Agile teams would have to use estimation checklist repeatedly on the same project with each iteration and/or release. We tried to include only those factors in each checklist, which were considered relevant by the team members. Even then, in Infoway case, we had to divide the checklist factors into mandatory and optional parts.
- Share results of analysis: It is also important to share the results of the data analysis with case participants. For example, if there is any improvement in the accuracy of the effort estimates. And, how the whole team thinks about the usefulness of the estimation checklist. Showing benefits of the proposed solutions to the industrial partner is a good practice to improve industry-academia collaboration [45].

6.4. Implications for practice

Expert estimation is most widely used estimation technique in software Industry. It involves little documentation and formalism. And, instead of using some algorithm or model, the involved experts rely on their intuition to estimate software development effort. Despite of all these "popular" attributes of expert estimation, it could still be inconsistent and ad-hoc. The experts may consider one set of factors during estimation one day, and a different set of factors another day. Furthermore, which factors were actually considered in a specific case may also gets lost. Checklist can be used as a tool to overcome these problems associated with expert estimation. On one hand, checklist would ensure the consistent consideration of the set of factors during effort estimation. On the other hand, it also documents, at low cost, what was considered during estimation of different stories and requirements. In three cases that we investigated in this study, the participants found the idea of using checklist during estimation beneficial. The involved companies are from different domains, and their size varies from small to medium.

As noted before as well, in order for checklist to be useful, it has to be customized for the involved team and company context. This can only be realized if the involved team actively works with the researchers to provide them their feedback, and use the checklist according to the provided guidelines. Apart from the qualitative information through interviews and discussions, effort estimation data (i.e., actual effort, effort estimates) is required to be able to perform quantitative analysis. For example, in order to see if the checklist has improved accuracy of the effort estimates, effort data for the past and current sprints/release is required.

The proposed process (see Figure 1) can be used by the managers to prepare customized checklists for their teams. They can start with arranging workshops or discussions with the practitioners involved in the estimation process to elicit the reasons for the lack of accuracy and consistency in their effort estimates. Through these discussions, an initial checklist consisting of the factors that should be considered during estimation can be prepared. It is crucial that the team develops a shared understanding of the checklist factors, and how to use them during effort estimation. Next, the involved team should test the checklist on trial basis in few sprints. This pilot testing is vital in understanding how to use and integrate the checklist in the estimation process, and also in revising them using team's feedback. We observed during these studies that the suggestion (cf. [31, 29]) to keep the checklists brief and relevant is crucial to ensure that the team members remain motivated in using the checklist. Lastly, it is also essential for the managers and teams to remember that the checklist should not be treated as a static document. In the light of their experiences and analysis, the checklist should continue to evolve.

7. Validity threats

In this section we discuss the validity aspects of our study using the classification adopted by Runeson and Höst [27] in their guidelines from Yin [46] and Wohlin et al [47].

7.1. Construct validity

Construct validity is about ensuring if the measures used actually represent the concepts that researchers want to investigate and what is described in the research questions. In our case, for example, we discussed two effort estimation scenarios (under and overestimation) and their possible reasons with interviewees. In order to make sure that the interviewees do not interpret them in a different way, we provided them with examples to explain what do we mean by under and overestimation (cf, scenarios used in written interview questions in Appendix A).

The proposed estimation checklist contain number of factors and corresponding scale values that are to be used by the study participants during estimation. We performed many steps to ensure that each checklist factor is understood and interpreted correctly and consistently by all team members. These steps include presenting each factor and associated scale in a workshop, written description notes for each factor to explain what is meant by it and how to use the associated scale, static validation and the follow up discussion to ensure shared understanding.

7.2. Internal validity

Internal validity relates to the possibility of confounding factors impacting the investigated factor, wherein the researcher is either unaware of the existence of these factors or their impact on the investigated factor. When we elicited the reasons for effort overruns, in each case we involved multiple participants (source triangulation) to ensure that we have not missed out anything. Furthermore, after dynamic validation step, the metrics data showed improvement in the accuracy of the effort estimates in two cases (Infoway and Diyatech cases). We conducted follow up discussions (method triangulation) to request the case participants to explain the results, and to see if the improvements could actually be attributed to some other factor. In both cases, the participants attributed the improvements to the use of checklists.

Regarding the checklist development process, we used an instrument (see data collection instrument C in Appendix A) to ensure that the checklist development process has also been transferred to the case companies. It is possible that the respondents found it difficult to acknowledge their lack of understanding of the process or checklist in response to the questions in the instrument i.e., there is a possibility of respondent bias. We tried to triangulate respondents' written feedback in the follow-up semi-structured interviews. Nonetheless, there is still a possibility that some respondents did not fully understood the checklist development process.

7.3. External validity

External validity is about the extent to which the study findings are generalizeable and applicable to cases and people outside the investigated case. We investigated three cases in this study. Our findings are limited to the type of cases we studied and their context. Two companies (Infoway and Diyatech) are medium sized relatively old companies, while the third one (TSoft) is a new and small offshore development center of a parent company in a European country. The companies are developing software in different domains (eHealth, performance enhancement and transportation). Two common aspects in these three studied cases are: expert estimation and agile practices. We can therefore say that our findings may be of interest to the small to medium software companies that use expert estimation to estimate development effort in an agile software development context.

7.4. Reliability

Reliability is concerned with the degree to which the data and the analysis are dependent on the researcher. We involved multiple researchers during the study planning, data collection and analysis phases. We also involved the case participants (company director for Infoway, development manager for Diyatech, project manager for TSoft) in reviewing the study protocol before the start of the data collection phase. However, since the work is lead by the first author (PhD student), we can not be certain that the data and analysis are completely independent of the first author. The involvement of other researchers in all study phases would have reduced this dependency, if not completely eliminated it.

We were able to obtain limited effort data both before and after the use of checklist. In case of TSoft, previous effort data was not available. Therefore, we could not compare in quantitative terms the impact of checklist on the accuracy of the effort estimates. Furthermore, in case of Diyatech we have data about two releases only, one without the checklist use and one where checklist was used. This is not enough to be able to draw any conclusions. In case of Infoway we have relatively more data, but still the results should be interpreted with care. However, it was good

to observe during the follow up interaction with these companies, that they are still using the checklists. We hope we will be able to have more data, both qualitative and quantitative, in future to be able to perform more analysis.

8. Conclusions and future work

We reported a multi-case study to develop and evolve estimation checklists to improve their expert estimation processes. The three companies involved in the reported multi-case study are: Infoway (a medium-sized company in Brazil developing products in eHealth domain), Diyatech (a medium-sized company in Pakistan developing performance enhancement applications for distributed environments) and TSoft (small and a new offshore development center of a parent company in Norway, developing applications in the transportation domain).

In **RQ1**, we proposed a multi-step process to develop and evolve checklist to improve expert estimation of software development effort. The application of the process was demonstrated in three companies. The checklists were developed and evolved based on the feedback from the participants in all cases. After dynamically validating the checklists in the real context, we transferred the revised checklists to the involved teams. We performed a follow-up step after a gap of six months in all cases. In case of Infoway and Diyatech, we got to know that the checklist has become a part of their estimation process. In case of TSoft, the development team was not using the checklist anymore. However, the project manager was using checklist factors to prioritize product backlog. We did not perform any follow up with Diyatech, as the checklist was transferred to them only recently after dynamic validation. Finally, we also ensured that the checklist development process is also transferred to the involved teams so that, if and when required, they can use it by themselves in developing new estimation checklists.

In **RQ2**, based on a combination of qualitative and quantitative data, we showed the usefulness of checklists in improving the expert effort estimation. After using the checklist in the real context, participants from all cases shared that they found the checklist extremely useful in improving the estimation process. The commonly observed benefits of estimation checklists were: improved task understandability, increased confidence in effort estimates, fewer chances of missing tasks and factors to be considered, and more objectivity in the expert estimation process. The analysis of the effort data from two cases showed that the use of checklist had improved the accuracy of the effort estimates, and the underestimation bias was also reduced, which was one of the common challenges for all three companies before the introduction of checklists.

As part of our future work, we plan to collect more data from these companies to see how useful have been the checklists in the long run. We will also try to identify possibilities for further improvements related to the checklists. We are currently using the process proposed in this study in a large Swedish telecommunication software vendor, which was interested in using our work to improve their expert estimation process.

Another future direction is about combining checklist with Machine Learning (ML) based techniques to calculate the effort estimate automatically once all checklist factors are processed, and relevant options are selected by the experts involved. Analogy-based estimation techniques such as Case Base Reasoning (CBR) are one of the most frequently used ML based techniques [1, 48, 49]. Analogy based estimation is more closely related to expert estimation, as it involves a problem solving mechanism that is quite similar to human problem solving [50, 51, 52, 53]. Therefore, experts are more likely to use it, in combination with expert estimation, as compared to other ML based estimation techniques. Once our checklist has been used for few sprints or releases, a useful historical database would become available. The idea is to use this database to support experts in generating estimates once they have answered all checklist factors. Analogy or CBR based techniques can use the historical data to identify similar estimation cases from past to derive or propose the effort estimate for the current case.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback, which has helped us in improving the paper.

References

- [1] M. Jørgensen, M. Shepperd, A systematic review of software development cost estimation studies, *IEEE Transactions on software engineering* 33 (1).
- [2] M. Jørgensen, A review of studies on expert estimation of software development effort, *Journal of Systems and Software* 70 (1) (2004) 37–60.

- [3] M. Usman, E. Mendes, 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, EASE'15, 2015, pp. 12:1–12:10.
- [4] B. W. Boehm, et al., Software engineering economics, Vol. 197, Prentice-hall Englewood Cliffs (NJ), 1981.
- [5] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, R. Selby, Cost models for future software life cycle processes: Cocomo 2.0, *Annals of software engineering* 1 (1) (1995) 57–94.
- [6] K. E. Wiegers, More about software requirements, Microsoft Press, 2006.
- [7] N. Harvey, Improving Judgment in Forecasting, Springer US, Boston, MA, 2001, pp. 59–80.
- [8] M. Usman, E. Mendes, F. Weidt, R. Britto, Effort estimation in agile software development: A systematic literature review, in: Proceedings of the 10th International Conference on Predictive Models in Software Engineering, PROMISE'14, 2014, pp. 82–91.
- [9] J. Cho, Issues and challenges of agile software development with scrum, *Issues in Information Systems* 9 (2) (2008) 188–195.
- [10] M. Jørgensen, D. Sjøberg, R. Conradi, Reuse of software development experience at telenor telecom software, in: Proc. European Software Process Improvement Conf, 1998, pp. 10–19.
- [11] K. M. Furulund, K. Moløkken-Østvold, Increasing software effort estimation accuracy using experience data, estimation models and checklists, in: Quality Software, 2007. QSIC'07. Seventh International Conference on, IEEE, 2007, pp. 342–347.
- [12] U. Passing, M. Shepperd, An experiment on software project size and effort estimation, in: Empirical Software Engineering, 2003. ISESE 2003. Proceedings. 2003 International Symposium on, IEEE, 2003, pp. 120–129.
- [13] B. M. Hales, P. J. Pronovost, The checklist - a tool for error management and performance improvement, *Journal of critical care* 21 (3) (2006) 231–235.
- [14] A. Gawande, The Checklist Manifesto: How To Get Things Right, Profile Books, 2010.
- [15] W. Y. Higgins, D. J. Boorman, An analysis of the effectiveness of checklists when combined with other processes, methods and tools to reduce risk in high hazard activities, *Boeing Technical Journal*.
- [16] D. Boorman, Today's electronic checklists reduce likelihood of crew errors and help prevent mishaps, *ICAO Journal* 56 (1) (2001) 17–22.
- [17] D. Boorman, Safety benefits of electronic checklists: An analysis of commercial transport accidents, in: Proceedings of the 11th International Symposium on Aviation Psychology. Columbus, USA., 2001.
- [18] P. Pronovost, Interventions to decrease catheter-related bloodstream infections in the icu: The keystone intensive care unit project, *American Journal of Infection Control* 36 (10) (2008) S171.e1 – S171.e5.
- [19] P. Pronovost, D. Needham, S. Berenholtz, D. Sinopoli, H. Chu, S. Cosgrove, B. Sexton, R. Hyzy, R. Welsh, G. Roth, et al., An intervention to decrease catheter-related bloodstream infections in the icu, *New England Journal of Medicine* 355 (26) (2006) 2725–2732.
- [20] P. Pronovost, E. Vohr, Safe patients, smart hospitals: how one doctor's checklist can help us change health care from the inside out, Penguin, 2010.
- [21] A. Latif, M. S. Halim, P. J. Pronovost, Eliminating infections in the icu: Clabsi, *Current Infectious Disease Reports* 17 (35).
- [22] O. Laitenberger, J.-M. DeBaud, An encompassing life cycle centric survey of software inspection, *Journal of Systems and Software* 50 (1) (2000) 5 – 31.
- [23] S. Ebad, Inspection reading techniques applied to software artifacts - a systematic review, *Computer Systems Science and Engineering* 32 (3) (2017) 213–226.
- [24] E. Teixeira, R. De Mello, R. Motta, C. Werner, A. Vasconcelos, Verification of software process line models: A checklist-based inspection approach, 2015, pp. 81–94.
- [25] R. Maiani de Mello, E. Nogueira Teixeira, M. Schots, C. Lima Werner, G. Travassos, Verification of software product line artefacts: A checklist to support feature model inspections, *Journal of Universal Computer Science* 20 (5) (2014) 720–745.
- [26] F. Elberzhager, A. Klaus, M. Jawurek, Software inspections using guided checklists to ensure security goals, 2009, pp. 853–858.
- [27] P. Runeson, M. Höst, A. Rainer, B. Regnell, Case Study Research in Software Engineering: Guidelines and Examples, John Wiley & Sons, 2012.
- [28] B. A. Kitchenham, D. Budgen, P. Brereton, Evidence-based software engineering and systematic reviews, Vol. 4, CRC Press, 2015.
- [29] B. Hales, M. Terblanche, R. Fowler, W. Sibbald, Development of medical checklists for improved quality of patient care, *International Journal for Quality in Health Care* 20 (1) (2007) 22–30.
- [30] M. Scriven, The logic and methodology of checklists, accessed: Jan 12, 2018 (2000).
URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.588.7093&rep=rep1&type=pdf>
- [31] M. Jørgensen, K. Moløkken, A preliminary checklist for software cost management, in: Quality Software, 2003. Proceedings. Third International Conference on, IEEE, 2003, pp. 134–140.
- [32] M. Usman, J. Börstler, K. Petersen, An effort estimation taxonomy for agile software development, *International Journal of Software Engineering and Knowledge Engineering* 27 (04) (2017) 641–674.
- [33] K. Moløkken, M. Jørgensen, A review of software surveys on software effort estimation, in: Empirical Software Engineering, 2003. ISESE 2003. Proceedings. 2003 International Symposium on, IEEE, pp. 223–230.
- [34] M. Usman, R. Britto, L.-O. Damm, J. Börstler, Effort estimation in large-scale software development: An industrial case study, *Information and Software Technology* 99 (2018) 21–40.
- [35] M. B. Miles, A. M. Huberman, Qualitative data analysis: An expanded sourcebook, sage, 1994.
- [36] M. Shepperd, M. Cartwright, G. Kadoda, On building prediction systems for software engineers, *Empirical Software Engineering* 5 (3) (2000) 175–182.
- [37] B. A. Kitchenham, L. M. Pickard, S. G. MacDonell, M. J. Shepperd, What accuracy statistics really measure, *IEE Proceedings-Software* 148 (3) (2001) 81–85.
- [38] M. Jørgensen, A critique of how we measure and interpret the accuracy of software development effort estimation, in: First International Workshop on Software Productivity Analysis and Cost Estimation, Citeseer, 2007.
- [39] V. Mahnič, T. Hovelja, On using planning poker for estimating user stories, *Journal of Systems and Software* 85 (9) (2012) 2086–2095.
- [40] K. Moløkken-Østvold, N. C. Haugen, H. C. Benestad, Using planning poker for combining expert estimates in software projects, *Journal of Systems and Software* 81 (12) (2008) 2106–2117.

- [41] H. B. Mann, D. R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, *The annals of mathematical statistics* (1947) 50–60.
- [42] T. Gorschek, P. Garre, S. Larsson, C. Wohlin, A model for technology transfer in practice, *IEEE software* 23 (6) (2006) 88–95.
- [43] S. Grimstad, M. Jorgensen, K. Molokken-Ostfold, The clients' impact on effort estimation accuracy in software development projects, in: *Software Metrics*, 2005. 11th IEEE International Symposium, IEEE, 2005, pp. 3:1–3:10.
- [44] S. Keaveney, K. Conboy, Cost estimation in agile development projects., in: *ECIS*, 2006, pp. 183–197.
- [45] V. Garousi, K. Petersen, B. Ozkan, Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review, *Information and Software Technology* 79 (2016) 106 – 127.
- [46] R. K. Yin, *Case Study Research: Design and Methods*, Vol. 2003.
- [47] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén, *Experimentation in software engineering*, Springer Science & Business Media, 2012.
- [48] A. Idri, M. Hosni, A. Abran, Improved estimation of software development effort using classical and fuzzy analogy ensembles, *Applied Soft Computing* 49 (2016) 990 – 1019.
- [49] J. Wen, S. Li, Z. Lin, Y. Hu, C. Huang, Systematic literature review of machine learning based software development effort estimation models, *Information and Software Technology* 54 (1) (2012) 41 – 59.
- [50] J. Li, G. Ruhe, A. Al-Emran, M. M. Richter, A flexible method for software effort estimation by analogy, *Empirical Software Engineering* 12 (1) (2007) 65–106.
- [51] L. Angelis, I. Stamelos, A simulation tool for efficient analogy based cost estimation, *Empirical software engineering* 5 (1) (2000) 35–68.
- [52] H. K. Leung, Estimating maintenance effort by analogy, *Empirical Software Engineering* 7 (2) (2002) 157–175.
- [53] J. Keung, Software development cost estimation using analogy: A review, in: *2009 Australian Software Engineering Conference*, 2009, pp. 327–336. doi:10.1109/ASWEC.2009.32.

Appendix A. Checklist versions and data collection instruments

Table A.1: Infoway checklist V2 (after initial feedback and static validation)

No.	Checklist factor	Scale
1	What is the type of the task?	Implementation, testing, research
2	The task is implementing a Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	Only an NFR, FR with major NFR constraints, FR with minor NFR constraints, only FR
3	How similar is the task to the previously developed tasks	To a great extent, somewhat, very little, not at all
4	Rank team's technical competence/skills required to implement and test this task	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this task	Very good, good, average, below average, lacking
6	Does the task require an architectural change?	Yes, not sure, no
7	Does the task involve communication between multiple (sub) systems?	Yes, not sure, no
8	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
9	How clear is your understanding of the task?	Very good, good, average, below average, need to talk to customer before moving on
10	Is the task size suitable to fit in the planned sprint	Yes, not sure, no
11	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no
12	Does the team has new member(s)?	Yes, no
13	Rank team's recent productivity	Very good, good, average, below average, lacking
Based on the above characterizations, most likely effort estimate for this task is: _____ hours		

Data Collection Instrument A

Interview questions about effort estimation at Diyatech

Please provide information about your role and experience in Table 1

Table 1: Basic information of the interviewee

Your current role	Software Architect
Experience in current role (in Years)	
Previous roles in this company, if any	
Total experience in this company	
Experience before this company, if any	

There are two common scenarios (see Table 2) with respect to the accuracy of the effort estimates: underestimation (Scenario 1) and overestimation (Scenario 2). Underestimation is also referred as effort overrun or delay.

Table 2: Effort estimation example scenarios

Scenario 1 (underestimation)	Scenario 2 (overestimation)
Effort estimate: 25 Hours	Effort estimate: 25 Hours
Actual effort: 40 Hours	Actual effort: 15 Hours

1. How frequently these two scenarios occur in your product? Which one is more frequent?
2. In your opinion, what are the main reasons for software effort overruns, i.e. scenario 1, in your context?
3. In your opinion, what are the main reasons for overestimation scenario, i.e. scenario 2, in your context?
4. What are the main challenges/difficulties in estimating the development effort?
5. How effort is estimated in your product/company? Briefly describe main steps.
6. Which factors do you consider in estimating the effort of a requirement or story?
7. What role, if any, do you play in the effort estimation process?
8. How to improve the current effort estimation process? What changes should be made, in your opinion, in the current effort estimation process?

Developers are only asked Q1 to Q3, as they are not directly involved in the estimation process. The architect and the development manager were requested to answer all questions, as they participate in the estimation process.

Data Collection Instrument B

Effort Estimation Checklist Feedback Instrument

First of all, we reproduced the relevant checklist here to facilitate the participants in answering feedback questions.

Evaluation of the proposed effort estimation checklist by practitioners

1. Questions about each checklist item

Provide your opinion on each checklist factor with respect to their relevance to the effort estimation. Mark your choice by writing "Yes" in the cell of your choice.

#	Do you think this checklist item is relevant to consider during estimation?				
	Strongly agree	Agree	Undecided	Disagree	Strongly disagree
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

2. Suggestions for improvement (This part is really important, as your suggestions will guide us in improving the proposed checklist)

2.1 Are there any other factors that should be added to the checklist? If yes, then kindly list those factors here, also state why do you recommend adding them.

Space for the answer

2.2 Are there checklist items that should be removed from the checklist? If yes, then kindly list those items here and also state why do you recommend removing them?

Space for the answer

2.3 What are the benefits of using this estimation checklist?

Space for the answer

2.4 What are the drawbacks/limitations of this checklist in your opinion? How can we improve/revise this checklist? Please feel free to provide your frank opinion, as it will help us in improving our work.

Space for the answer

31

2.5 Are you interested in using the checklist in future sprints/releases?

Space for the answer

Data Collection Instrument C

Checklist Development Process Feedback

Earlier we obtained your feedback on the checklist. In this part, we are requesting your feedback and reflections **on the process** we used to develop and handover checklist to your team.

1. Checklist development process (included here for your reference)

Figure 1 from the Manuscript was included here to present a summary of checklist development process

2. Questions

Q1: Do you understand all activities (1 to 5 in the Figure above) of the process and their purpose?

Q2: Do you think you can apply this process by yourself in developing a checklist for a different team/project/context? If no, are there any activities that you think are difficult to apply?

Q3: Is there a need to remove any activities from this process? If yes, then kindly list those activities here and also state why do you recommend removing them?

Q4: Is there a need to add activities to this process? If yes, then kindly list those activities here and also state why do you recommend adding them?

Q5: You were closely involved when this process was applied to develop checklist for your team. Do you think this process was helpful in developing checklist for your team?

Q6: Do you have any other experiences to share or reflections/suggestions to improve it further?

Table A.2: Infoway checklist V3 (prioritized and classified after dynamic validation)

No.	Checklist factor	Scale
Mandatory Part		
1	What is the type of the task?	Implementation, testing, research
2	The task is implementing a Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	Only an NFR, FR with major NFR constraints, FR with minor NFR constraints, only FR
3	Rank team's technical competence/skills required to implement and test this task	Very good, good, average, below average, lacking
4	How similar is the task to the previously developed tasks?	To a great extent, somewhat, very little, not at all
5	Rank team's domain knowledge required to implement this task	Very good, good, average, below average, lacking
6	How clear is your understanding of the task?	Very good, good, average, below average, need to talk to customer before moving on
7	The implementation of the task requires understanding and/or changing legacy code	Yes, not sure, no
8	Does the task involve communication between multiple (sub) systems?	Yes, not sure, no
Optional Part		
9	Does the task require an architectural change?	Yes, not sure, no
10	Does the team has new member(s)?	Yes, no
11	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no
12	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
13	Is the task size suitable to fit in the planned sprint?	Yes, not sure, no
14	Rank team's recent productivity	Very good, good, average, below average, lacking
Based on the above characterizations, most likely effort estimate for this task is: _____ hours		

Table A.3: Diyatech checklist V2 (after initial feedback and static validation)

No.	Checklist factor	Scale
1	The requirement is classified as:	x, y, z (company specific categories)
2	How similar is the requirement to the previously developed requirements?	To a great extent, somewhat, very little, not at all
3	Rank team's technical competence required to implement this requirement	Very good, good, average, below average, lacking
4	Rank team's technical competence required to test this requirement	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this requirement	Very good, good, average, below average, lacking
6	Rank team's recent productivity	Very good, good, average, below average, lacking
7	Does the team has new member(s)?	Yes, no
8	Will this requirement's implementation involve an architectural change?	Yes, not sure, no
9	Will this requirement's implementation involve communication between multiple (sub) systems?	Yes, not sure, no
10	The requirement is classified as Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	Only an NFR, FR with major NFR constraints, FR with minor NFR constraints, only FR
11	How clear is your understanding of the requirement?	Very good, good, average, below average, need to talk to customer before moving on
12	Is the requirement size suitable to fit in the planned release?	Yes, not sure, no
13	What is the priority of the requirement?	Company specific priority categories
Based on the above characterizations, specify your most likely effort estimate for the following activities:		
	Requirements analysis	_____ hrs
	Documentation	_____ hrs
	Design	_____ hrs
	Coding	_____ hrs
	Testing	_____ hrs
	Non-development work	_____ hrs
Total		_____ hrs

Table A.4: Diyatech checklist V3 (after dynamic validation)

No.	Checklist factor	Scale
1	The requirement is classified as:	x, y, z (company specific categories)
2	How similar is the requirement to the previously developed requirements?	To a great extent, somewhat, very little, not at all
3	Rank team's technical competence required to implement this requirement	Very good, good, average, below average, lacking
4	Rank team's technical competence required to test this requirement	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this requirement	Very good, good, average, below average, lacking
6	Rank team's recent productivity	Very good, good, average, below average, lacking
7	Does the team has new member(s)?	Yes, no
8	Will this requirement's implementation involve an architectural change?	Yes, not sure, no
9	Will this requirement's implementation involve communication between multiple (sub) systems?	Yes, not sure, no
10	The requirement is classified as Functional Requirement (FR) or Non-Functional Requirement (NFR) or both?	Only an NFR, FR with major NFR constraints, FR with minor NFR constraints, only FR
11	How clear is your understanding of the requirement?	Very good, good, average, below average, need to talk to customer before moving on
12	Is the requirement size suitable to fit in the planned release?	Yes, not sure, no
13	What is the priority of the requirement?	Company specific priority categories
14	Will this requirement's implementation involve designing and/or revising and/or implementing distributed algorithm(s)?	Yes, not sure, no
15	Do you think there may be interruptions during the planned release/sprint affecting team member's momentum due to context switch?	Yes, not sure, no

Based on the above characterizations, specify your most likely effort estimate for the following activities:

Requirements analysis	_____ hrs
Documentation	_____ hrs
Design	_____ hrs
Coding	_____ hrs
Testing	_____ hrs
Non-development work	_____ hrs
Total	_____ hrs

Table A.5: Tsoft checklist V2 (after initial feedback and static validation)

No.	Checklist factor	Scale
1	What is the type of the task?	UI, Business logic, DB, UI and Business logic, Business logic and DB, All
2	How clear is your understanding of the task?	Very good, good, average, below average, lacking, need to talk to product owner before moving on
3	How similar is the task to the previously developed tasks	To a great extent, somewhat, very little, not at all
4	Rank team's technical competence/skills required to implement and test this task	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this task	Very good, good, average, below average, lacking
6	Will the task implementation likely to involve interfacing with external services/systems?	Yes, not sure, no
7	Does the task require an architectural change?	Yes, not sure, no
8	Does the task involve communication between multiple (sub) systems?	Yes, not sure, no
9	Does the task involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
10	Are there stringent non-functional requirements (performance, security etc.) associated with this task?	Yes, not sure, no
11	Will the team or some member(s) be working on other products/projects/tasks in parallel?	Yes, not sure, no
12	Is the task size suitable to fit in the planned sprint	Yes, not sure, no

Based on the above characterizations, most likely effort estimate for this task is: _____ hours

Table A.6: Tsoft checklist V3 (after follow up discussion with project manager)

No.	Checklist factor	Scale
1	What types of tasks are likely to be involved for this backlog item ?	UI, Business logic, DB, UI and Business logic, Business logic and DB, All
2	How clear is your understanding of the backlog item?	Very good, good, average, below average, lacking, need to talk to product owner before moving on
3	How similar is the item to the previously developed items	To a great extent, somewhat, very little, not at all
4	Rank team's technical competence/skills required to implement and test this backlog item	Very good, good, average, below average, lacking
5	Rank team's domain knowledge required to implement this backlog item	Very good, good, average, below average, lacking
6	Does the implementation of the backlog item likely to involve interfacing with external services/systems	Yes, not sure, no
7	Does the implementation of the backlog item likely to require an architectural change?	Yes, not sure, no
8	Does the implementation of the backlog item likely to involve communication between multiple (sub) systems?	Yes, not sure, no
9	Does the implementation of the backlog item likely to involve accessing and/or modifying several different elements/tables in the persistence/DB layer?	Yes, not sure, no
10	Are there stringent non-functional requirements (performance, security etc.) associated with this backlog item?	Yes, not sure, no
11	Will all or some member(s) of team be working on other products/projects/tasks in parallel?	Yes, not sure, no
12	Does the size of the backlog item suitable to fit in one sprint	Yes, not sure, no
Based on the above characterizations, most likely effort estimate for this task is: ——— hours		