Operationalizing the Experience Factory for Effort Estimation in Agile Processes

D. Taibi, V. Lenarduzzi
University of Bolzano/Bozen
Bolzano/Bozen
Italy
{dtaibi; vlenarduzzi}@unibz.it

P. Diebold
Fraunhofer IESE
Kaiserslautern
Germany
philipp.diebold@iese.fraunhofer.de

I. Lunesu
University of Cagliari
Cagliari
Italy
ilaria.lunesu@diee.unica.it

ABSTRACT

[Background] The effort required to systematically collect historical data is not always allocable in agile processes and historical data management is usually delegated to the developers' experience, who need to remember previous project details. However, even if well trained, developers cannot precisely remember a huge number of details, resulting in wrong decisions being made during the development process. [Aims] The goal of this paper is to operationalize the Experience Factory in an agile way, i.e., defining a strategy for collecting historical project data using an agile approach. [Method] We provide a mechanism for understanding whether a measure must be collected or not, based on the Return on Invested Time (ROIT). In order to validate this approach, we instantiated the factory with an exploratory case study, comparing four projects that did not use our approach with one project that used it after 12 weeks out of 37 and two projects that used it from the beginning. [Results] The proposed approach helps developers to constantly improve their estimation accuracy with a very positive ROIT of the collected measure. [Conclusions] From this first experience, we can conclude that the Experience Factory can be applied effectively to agile processes, supporting developers in improving their performance and reducing potential decision mistakes.

KEYWORDS

Knowledge Management, Experience Factory, Agile Software Development

1 INTRODUCTION

How can we systematically learn from past experience in agile projects without wasting too much time on collecting historical data? In agile processes, learning from past experience is one of the major concepts. Retrospective meetings are examples of learning moments where developers try to understand previous mistakes and discuss project issues. Developers usually remember and compare similar projects or tasks they developed in the past in order to avoid making previous mistakes, or adopt successful approaches. However, this comparison process is usually based on the developers' memory of past experience since the systematic collection of historical project data would take too much time and reduce the project speed. Therefore, an approach that supports effective and effortless collection of historical data would be of great benefit from a process improvement point of view.

In this paper, we want to fill this gap by introducing the concept of an "Agile Experience Factory", an agile approach for collecting historical data aimed at optimizing process efficiency and reducing errors due to wrong decisions. Our idea is to define an approach for storing only as little information as possible, with the goal of maximizing the benefits from the learning aspects and thus reducing data collection time as much as possible.

In order to test the Agile Experience Factory, in this paper we also report on an exploratory case study we conducted with seven projects, operationalizing the experience factory with the experience collected during the effort estimation process, with the goal of reducing the effort estimation overhead by learning from previous projects. We analyzed three projects developed without the Agile Experience Factory, one project that adopted it after 12 weeks out of 37, and two projects that adopted it from the beginning.

The results show that the proposed approach helped our developers improve effort estimation accuracy by spending a very limited amount of time on the collection phase (less than two minutes per user story) that highly justify the tiny effort required to collect the data. We believe that the process can also be applied to different project aspects by means of collecting different project data, such as for estimating a project's complexity or for understanding project issues.

This paper is structured as follows. Section 2 introduces the background and related work. Section 3 presents our idea of an Agile Experience Factory. Section 4 presents an instantiation of the factory with a case study on effort estimation to validate our proposal. Section 5 discusses on the results. Section 6 reports on threats to validity while finally, Section 7 draws conclusions and highlights future work.

2 BACKGROUND AND RELATED WORKS

In this section, we present work on different aspects that are related to our work in this paper. We report on how experience is captured and shared in agile processes and on the on current implementations of experience factories. Then we will introduce related work regarding the effort estimation approaches used in agile processes.

2.1 Knowledge Transfer in Agile

Knowledge represents the individuals' experiences, attitudes, and the context in which they work. Taking into account existing approaches or frameworks dealing with experience or knowledge sharing in agile environments, approaches range from informal communications [17] via comparison with other methods [19] to possible tool support for sharing knowledge[20]. Since communication is a major characteristic of agile development, most of the knowledge sharing aspects rely on it. Nonetheless, there are also other aspects such as the definition of a specific backlog of issues identified in the sprint retrospectives. But what all these approaches are missing and what differentiates our approach from them is the specific goal orientation combined with a kind of experience base of historical data for continuous learning.

Another approach for knowledge transfer is the Agile Practice Impact Model (APIM) [8] proposes an approach for storing experience regarding the impact of agile practices and thus could be combined with our Agile Experience Factory. This model is also a specific instantiation of the experience factory to some extent, but with a completely different focus than the Agile Experience Factory idea. The APIM focuses more on high-level process improvement of not-agile projects based on knowledge about the practices' impact, whereas our idea is more on the measurement level in projects that are already agile. However, both aspects could be combined to come up with a complete model for collecting historical measurement data, relating them to the impact of agile practices, and making improvement suggestions for their current (agile) process. In their work Ajmal and Koskinen [28] investigates the process of knowledge transfer in project-based organizations from the perspective of organizational culture. They identified constraint in time and budget and a lack of leadership as obstacles for knowledge transfer, they suggest to foster an organizational culture that facilitates and encourages the creation, sharing, and utilization of knowledge and ensuring that knowledge is produced and diffused across project boundaries to improve knowledge transfer. Our approach is different from this mentioned because their attention is focused on the organizational aspect instead of the use of specific tools or practices [21][22][28]. Chau and Maurer show Agile practices as a possible way to face the issue related to miscommunication within teams. In particular, they argued how to overcome the restriction in co-location and support organizational inter-team learning while supporting the social context critical to the sharing of tacit knowledge [26]. In this case, the communication aspect is crucial and they propose to use a lightweight and integrated knowledge sharing environment to overcome the limitation of to the various knowledge-sharing practices used by agile teams. The difference from our approach is represented by the use of a specific tool for helping team members' communication and knowledge sharing instead of a kind of experience repository to support organizations in understanding which technology can be effectively applied in the current agile process. Chau et al. also present a comparative analysis of knowledge sharing approaches of agile (e.g. XP and Scrum) and Tayloristic (traditional) software development teams. They also support the Experience Factory concept of reusing previous project experiences and a centralized knowledge management organization provides the infrastructure necessary in supporting continuous learning at the project team and organization levels [27].

2.2 The Experience Factory

The Experience Factory provides a mechanism for experience transfer, based on the Quality Improvement Paradigm (QIP) [9]. The Experience Factory has been defined as "a logical and/or physical organization that supports project developments by analyzing and synthesizing all kinds of experience, acting as a repository for such experience, and supplying that experience to various project on demand"[7]. The QIP, with its focus on continuous learning, really reflects the idea of agile development and is also the basic methodological framework for the Agile Experience Factory. To implement such learning or improvement, it is necessary to easily understand, modify, package, and learn from previous/historical experiences, which is represented in some of the main QIP steps:

Characterize: understand the environment based on available models, understand the existing organization processes and their criticalities

Set Goals: based on the organization's capabilities, set quantifiable goals for successful project improvement

Choose Process: select the development process

Execute: perform the process and collect the metrics identified in the "set goals" step

Analyze: at the end of the project, analyze the collected data, so as to evaluate the goals

Package: consolidate the experience gained, so as to make it available for future processes.

The idea of a Experience Factory started in the late 1980s with Basili and Rombach [5] and refined in 1991 [6] and in 1994 [7].

The Experience Factory has also been refined recently, in the domain of software process improvement (SPI) and applied to Agile processes, starting with [1] and evolving into the "Process Configuration Framework" (PCF) [2] as well as a specific instantiation in the avionics domain [3][4].

The same concept of the experience factory has been used in our previous work where we instantiated the factory collecting historical data on effort estimation, both investigating the effectiveness of data-driven[33][32] and expert-based[35] estimation models. Moreover, in our preliminary work we also instantiated the factory collecting data for different purposes such as the adoption of new processes [34], identifying reliability measures [36] and testing[37].

However, coming up with ideas of different experience factories is, to some extent, very easy when working on a high level. Nonetheless, implementing such an experience factory and instantiating it with at least some data is the harder part, which is most often omitted due to time and effort constraints. However, the most common reason why such experience factories do not become a reality is the aspect of easy data collection and storage. For this reason, we define the concept of Agile Experience Factory as an instantiation of the classical Experience Factory [7] but with a lightweight approach for packaging and using the collected data.

2.3 Effort Estimation in Agile

Effort estimation, in any software project, is a very critical task [21]. The success or failure of projects depends heavily on the accuracy of effort and schedule estimations. Also for agile software development, effort estimation is a big challenge producing inaccurate estimates [30]. Ziauddin et al. proposed a data-driven effort estimation model calibrated with data collected from many software projects showing a good estimation accuracy in terms of Mean Magnitude of Relative Error(MMRE) and PRED(n) (percentage of estimates for which magnitude of relative error (MRE) is lower or equal to n)[25]. However, this could be used as a generic prediction model but, as any model trained on different data, could not be perfectly fit to any project. Rashmi et al. focused their attention on understanding the causes of inaccurate estimates in Agile software development and problems related to estimation in Agile practices thereby proposing a method for accurate cost and effort negotiation [24].
Menzies et al. investigated if new SEE methods generated better estimates than older methods and they recommend to use COCOMO to generate predictions [23].
Coelho et al. offered an overview of the different size estimation methods traditionally used such as Story Points [22].
Tanveer et al. investigate estimation process with respect to its accuracy in the context of agile software development from the perspective of agile development teams reporting that the accuracy of estimation depends also on developer's knowledge and experience and there is a need for a tool that incorporates expert knowledge, enabling explicit consideration of cost drivers by experts and visualizing this information in order to improve the effectiveness of the effort estimation process. In our Agile Experience Factory, this approach is implemented and extended.
Lenarduzzi et al investigated if functional size measurement could improve the effort estimation in Scrum, reporting that expert-based effort estimation, even if inaccurate, is still more reliable than those based on functional measures [14][15].
Most of the aforementioned works have in common the study of inaccurate estimates through the use of tools or analytic methods and techniques [30]. Accurate estimation requires a multidimensional view to produce accurate and effective estimates so it is proposed a model developed to accommodate most of the characteristics of Agile methodology, especially Adaption and Iteration [25] or one, starting from limitations of existing methods, customized for agile estimations [24].
What mainly differentiates our approach from all of them is the specific goal orientation and the approach to face the Agile estimation issues. Instead, a methodology to collect historical data to verify and understand whether collecting a certain piece of information promises to be effective, some models, size estimation methods [22] or techniques based on experience are proposed [29].

3 THE AGILE EXPERIENCE FACTORY

In this section, we will describe our proposal of an Agile Experience Factory based on an agile approach for packaging and using the collected historical data.
The Agile Experience Factory is an approach intended to help software development organizations synthesize different experiences. It acts as an experience repository to support organizations in understanding which technology can be effectively applied in the current agile process, without requiring any noticeable effort to obtain the information or to store the collected information.
The idea is based on the effective, goal-driven collection of historical data in order to check and understand whether collecting a certain piece of information promises to be effective. We identify the effectiveness of the collection of a piece of information based on the Return on Invested Time (ROIT), as an extension of the classical definition of the Return on Investment (ROI), calculated as

$$\text{ROI} = \frac{\text{gain from investment} - \text{cost of investment}}{\text{cost of investment}}$$

$$\text{ROIT} \cong \frac{\text{saved time} - \text{collection time}}{\text{collection time}}$$

The saved time can be considered as time saved thanks to the right decision made with the help of the collected information, while the collection time is the time needed to collect, store, and retrieve the metric information. Thus, we generally define as effective metrics those that generate a ROIT larger than zero. For example, an effective measure that required a collection time of one hour will allow saving more than two hours thanks to the knowledge acquired with the related data. If the measure provides an ROIT close to zero, we suggest stopping its collection and reporting to the Agile Experience Factory that this measure is not effective in the specific context.
In some cases, the "saved time" is not applicable, and other measures should be considered. As example, in the case of effort estimation, a higher estimation accuracy would not help to save any time but to reduce the error and the likelihood of overrunning costs. Therefore, the ROIT should be intended as a usefulness measure with the goal of understanding if the collected measure should be collected and not as the classical ROI definition, as intended in the economical domain.
Consequently, for such a selection of measures, it would be necessary to have a huge set of measures in the experience factory together with some additional context information.

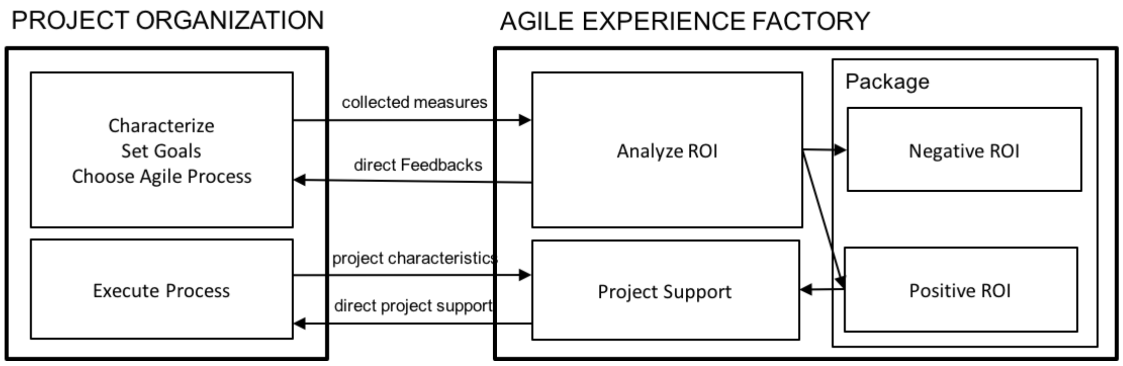

Figure 1: The Agile Experience Factory

For example, it is necessary to (1) categorize the different measures in order to find similar or precluding measures, (2) identify information on how to collect these measures, and (3) determine what this collection costs in terms of time and/or money. Only with this data will it be possible to improve the data collection and further measurements in the next project or even the next iteration.

The process suggested by the Agile Experience Factory, as depicted in Figure 1, can be summarized as follows.

Before the beginning of an agile development process, an organization characterizes the current projects and their goals, selecting the metrics to be collected during the process based on the GQM approach, but also considering the metrics that previously reported an ROIT higher than zero. Then they start the collection of the selected metrics during the execution of the project.

Here is where the Agile Experience Factory instantiates the concept of QIP and [9] differs from other implementations.

After a pre-defined number of sprints, the organization starts a "pre-packaging" step, evaluating the ROIT of the collected measures based on the metrics defined in the previous steps. In case of measures with a negative ROIT, they can decide whether to keep on collecting them until the next sprint or stop collecting in order to avoid wasting time. In the Agile Experience Factory, all measures must provide a positive ROIT during the current development process.

Once the project is finished, the metric collection process is completed and no extra effort will be required to package the experience for future projects.

In order to provide an example instantiation of the Agile Experience Factory, in the next section we report on a multiple case study based on several projects where we instantiated the factory with one metric, aiming to understand whether collecting effort-related historical data is effective in a Scrum process.

4 INSTANTIATING THE AGILE EXPERIENCE FACTORY: A CASE STUDY ON EFFORT ESTIMATION

The objective of our research is to understand if the systematic collection of historical data in agile processes can improve the process efficiency itself. For this purpose, we conducted an exploratory case study[31] among two sets of projects considering as an example metric the effort data collected for each user story.

In this section, we will present the goal, questions, and metrics for the case study. We will then describe the designs used for the case study, the measurement instruments, and the analysis results.

4.1 Study Goal

During the sprint planning meeting, the development team estimates the effort for the user stories in the product backlog based on their experience with implementing similar stories. Scrum, and agile approaches in general, do not prescribe a unit of measure for estimating the effort but suggest using a more abstracted measure. Common estimating methods include, for example, numeric sizing, t-shirt sizes, and story points.

We therefore assume that building and maintaining such kinds of experience totally depends on the developers' memory, which usually supports them in comparing similar user stories developed in the past. With this study, we aim at understanding if the Agile Experience Factory can, by means of complex data collection, support developers in easily increasing their accuracy without slowing down the development processes.

According to our expectations, we formulated the goal of the case study following the GQM approach [10] as:

Analyze the Scrum process
for the purpose of evaluating and comparing
from the point of view of software developers
with respect to estimation accuracy and estimation efficiency
in the context of Scrum process development.

We formulated our research questions as follows and further derived questions and metrics from them:
Q1: Does introducing an experience -base dataset (AEF) in Scrum processes improve estimation accuracy over time?
Q2: What do the developers think about the applicability of this approach?
Q3: Is there a positive Return On Invested Time (ROIT) as a result of collecting this measure?
Q3.1: How much effort is needed to reduce the estimation error?
Q4: Are teams using our approach statistically significant better in the estimation process compared to those not using it?

Q1 – Accuracy: measures, for a given user story, the difference between actual and estimated effort. In order to evaluate the research question, we need to identify a common measure for the effort. The measure must be comparable among different projects and different teams after the development. To measure user stories, we considered the usage of story points, but since they cannot be compared across different projects and teams, we decided to collect effort in numbers of hours as well. We defined the actual effort as the number of hours spent per user story. The estimated effort was estimated by the developers during the planning poker, while the actual effort was tracked by the developers and collected by the Scrum master during the sprint retrospective.
We calculated the accuracy of the effort prediction by means of the Magnitude of Relative Error (MRE) defined as:

$$\text{MRE}= \frac{|\, actual\ effort - estimated\ effort \,|}{actual\ effort}$$

H11: developers estimate effort with a lower MRE when they apply the Agile Experience Factory by means of an experience-based data set.

$H1_{11}$ = MRE(AEF) < MRE(expert)
$H0_{11}$ = MRE(AEF) ≥ MRE(expert)

Q2 - Applicability: What do the participants think about the applicability of the Agile Experience Factory? To answer this question, we collected the time overhead needed to get the opinions of the participants by means of a set of metrics derived from the Technology Acceptance Model [11], collecting the metrics listed below at the end of the project by means of a questionnaire. Here we report on the definition of the adopted measures and the questions we identified to evaluate them. All questions were evaluated based on a 5-point ordinal Likert scale with the following options: 1 = strongly disagree, 2 = disagree, 3 = neither agree nor disagree, 4 = agree, 5 = strongly agree.
Perceived usefulness: measures the degree to which the participant considers the approach useful for making project decisions:

- I am sure that I was able to estimate the effort of a user story better with the Agile Experience Factory support.
- I was able to find similar user stories to compare with the one I was evaluating.
- I was able to estimate the effort better but the time required for using the new approach is too much compared to its benefits.

Perceived understandability: measures the effort needed by the subject to understand the approach being used or whether the participants will need to exert little effort to understand the relationship with the system concepts:

- It was easy for me to understand how the approach works.

Perceived easiness: measures the degree to which the subject believes that he or she was able to make project decisions easier than without the approach:

- It was easy for me to implement the approach with the shared spreadsheet.
- I was able to make effort estimations with less effort.
- I was able to remember previous project estimations more accurately.

Self-efficacy by applying the technique:

- Because I didn't need to remember by heart all the previous estimations, it was easy for me to keep an overview of the project.
- The approach helped me to increase my prediction accuracy.

With regard to the metrics, we derived the following three hypotheses:

- H21: The participants will perceive the understandability of the Agile Experience Factory higher than the average value α. notations differently than the without

$H0_{21}$: μAEF ≤α
$H1_{21}$: μAEF > α

- H22: The participants will perceive the ease of use of the Agile Experience Factory higher than the average value α.

$H0_{22}$: μAEF ≤ α
$H1_{21}$: μAEF > α

- H23: The subjects' feedback regarding self-efficacy will be higher than the average value α.

$H0_{23}$: μAEF ≤ α
$H1_{23}$: μAEF > α

Q3 - ROIT: ROIT is positive when the time spent for the data collection is smaller than the improvement on the estimation error for the next sprint. As an example, ROIT is positive if spending one minute to store the data and retrieve them helps the developers to save at least two one minutes. In case of negative or neutral ROIT, the measure can be removed from the collected data as proposed in Section 3.
In case of effort estimation, we are not reducing the development time but only estimating it better or worst. Therefore, a better estimation can be anyway attributed to the ROIT even if, in this case we are not applying the classic definition of ROIT (hours spent vs. hours saved).

Q3.1 - Effort to reduce the estimation error: This question complements Q3 providing detailed results on the effort needed to collect store and retrieve the effort.
$H0_{31}$: = Effort (AEF) ≥ Effort (without)
$H1_{31}$ =: Effort (AEF) < Effort (without)

4.2 Study Design

The study was designed as a case study [12], comparing four projects that did not use the Agile Experience Factory with two projects that used it from the beginning and one project that started using it after 12 weeks out of 37. All projects were developed with Scrum or Moonlight Scrum [13], a slightly modified version of Scrum for part-time developers working during non-overlapping hours. Moreover, the projects are similar in terms of their development context, such as same number of participants or team members, duration, and applications developed (see Table 1).
The first four projects, which did not use the Agile Experience Factory, were analyzed in order to define a benchmark. The project where the approach was introduced while it was ongoing was analyzed in order to understand possible changes in an existing process. The last set of three projects was analyzed to confirm our hypotheses, and the developers were interviewed during the process in order to capture their opinions regarding the applicability of the approach.
Before the beginning of each project, the participants were informed about the study and about the usage of the collected data. Then, after they had given their approval, they filled out a pre-questionnaire in which they provided information about their background.
The effort measures were collected during each sprint meeting by the Scrum master, in a shared online spreadsheet. The data collected in the shared spreadsheet for all projects includes: #id, story name, sprint number, creation date, completion date, developer(s), estimated effort (hours), actual effort (hours), error (MRE). We decided to delegate data collection to the Scrum master in order to avoid adding extra tasks to the developers. During the retrospective meeting, the Scrum master collected the actual effort and presented a line chart showing the accuracy trend of estimation accuracy over time, as illustrated in Figure 4. The charts shown are not meant to replace the burndown charts commonly used in agile projects, but to complement them, providing a quick overview of their estimation accuracy over the remaining effort and tasks represented in the burndown charts. Moreover, in order to get feedback from the developers, for the three projects that used the Agile Experience Factory we distributed a questionnaire to our developers after the last retrospective meeting.

4.3 Study Context

The study was applied to seven projects, four not using the Agile Experience Factory (Table 1, projects 1-4) and three applying the Agile Experience Factory (projects 5-7), which were executed by different universities and research centers. All projects were based on the development of web applications, with different purposes. Moreover, all projects were developed with Scrum supported by a Kanban board to better visualize issues and developers' assignments, with master and Ph.D. students as developers in five projects and professional developers and post-doc researchers with at least three years of development experience in two projects. Sprint iterations were fixed at two-week intervals in all projects.
We selected projects with different sizes, duration, and numbers of issues in order to validate the Agile Experience Factory on a wide range of projects. Table 1 shows a summary of the main information about the analyzed projects. Project #5 was developed in 37 weeks, without the Agile Experience Factory for the first 12 weeks (a) and with the help of the Agile Experience Factory during the remaining 25 weeks (b).

4.4 Data Analysis

Here, we describe the procedure applied to analyze the data collected.
We first performed a descriptive analysis of data. Then, we analyzed the normality by means of a Shaphiro-Wilk test.
If results show a normal distribution, we performed a t-test with a confidence interval of 95% to test our hypotheses. Otherwise, if the data were not normally distributed, we performed a Wilcoxon Signed-Rank test with a significance level of 0.05 to test our hypotheses.
Considering Q2, the participants provided their opinions about the Agile Experience Factory with regard to the metrics Completeness, Understandability, Easiness, and Self-Efficacy. Since we asked the participants to evaluate the questions based on a 5-point ordinal Likert scale (see Table 3 for more details on the scale), we consider values below 3 as negative, 3 as neutral, and values higher than 3 as positive. To make sure that the statements on the given scale were measuring the same underlying assumption, we performed a reliability test by calculating the Cronbach's α reliability measure. Cronbach's α is a common measure of internal consistency. It varies between 0 and 1 (values close to 1 indicate high internal consistency while values close to 0 indicate low internal consistency).

4.4 Study Results

For the first research question (Q1), we report the descriptive statistics of the project data in Table 1. Projects developed without the Agile Experience Factory show stable and common accuracy regarding effort estimation, with a Mean Magnitude of Relative Error (MMRE) that ranges from 40% to 70% (Table 1, Figure 2 and Figure 5). Even a project that lasted for three years (project #4), does not show any improvement over time.
The analysis of project #5, where we applied the approach after the sixth sprint (12 weeks), shows a constant MRE before the introduction of the Agile Experience Factory, and continuous improvements starting from the sprint after the introduction of the approach, with an MRE that improves from an initial value (see Figure 4). Moreover, one of the most important

achievements is the reduction of peaks in MRE, with a maximum value of 900%, compared to values of 430% reported in projects without the Agile Experience Factory.

The results obtained from the analysis of the projects that collected the effort data from the beginning show a dramatic improvement on MMRE (see Table 1 and Figure 3). The lines in Figure 4 represent the MMRE for each sprint.

Data collected for all seven projects is statistical significant with a $p<0.05$; therefore, based on the aforementioned results we can reject the null hypothesis (H01).

Taking into account **Q2**, we first verified the internal consistency by mean of the Cronbach's α measure. For all the projects, good to very good internal consistency was obtained for all metrics (Table 2). Then, we proceeded by aggregating the results obtained for $_{\text{each}}$ statement, calculating the average and median to arrive at the final result. By doing this, we were able to compare the results obtained for each metric (Table 3).

As reported in Table 3, the results show high applicability of the approach with regard to the collected effort measure. However, we must take into account that the questionnaire was applied only to three projects with a total of 14 participants.

As also confirmed by the Wilcoxon Signed-Rank, results obtained were statistically different from 3 thus rejecting the null hypothesis.

The analysis of Q3 shows the high ROIT of the collected measure. The approach allowed developers to reduce estimation error by an average of 72 minutes per user story (8% per sprint on average), while the time spent for collecting the data on each story and retrieving it when needed ranged from 1.5 to 3 minutes per user (Q3.1) confirming that the Agile Experience Factory requires more effort for the data-collection and retrieval part, even if irrelevant.

Therefore, this result confirms the usefulness of this measure and suggests continuing to collect this measure for future projects.

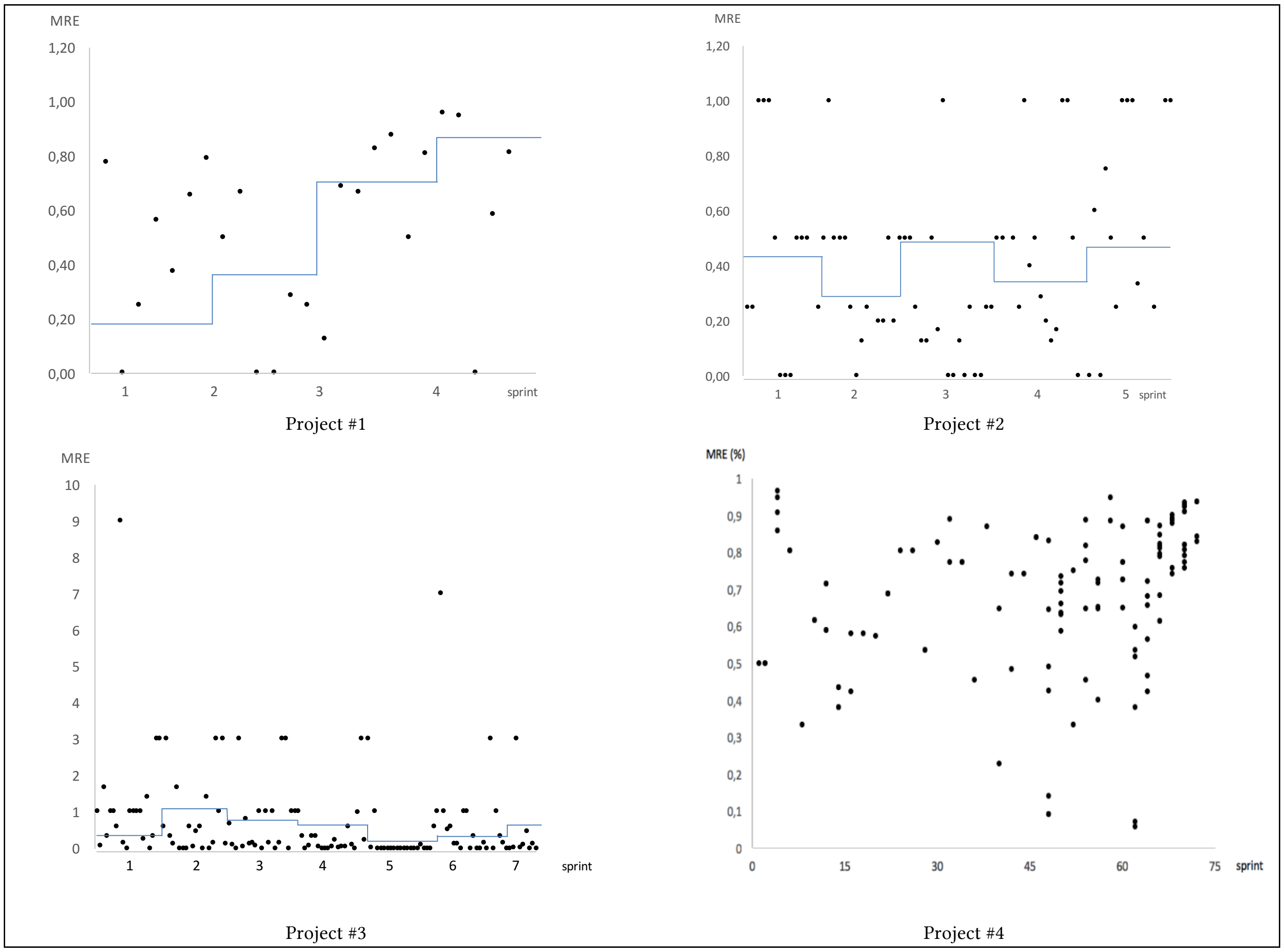

Figure 2: Estimation Accuracy (Q1) – Project NOT using the Agile Experience Factory

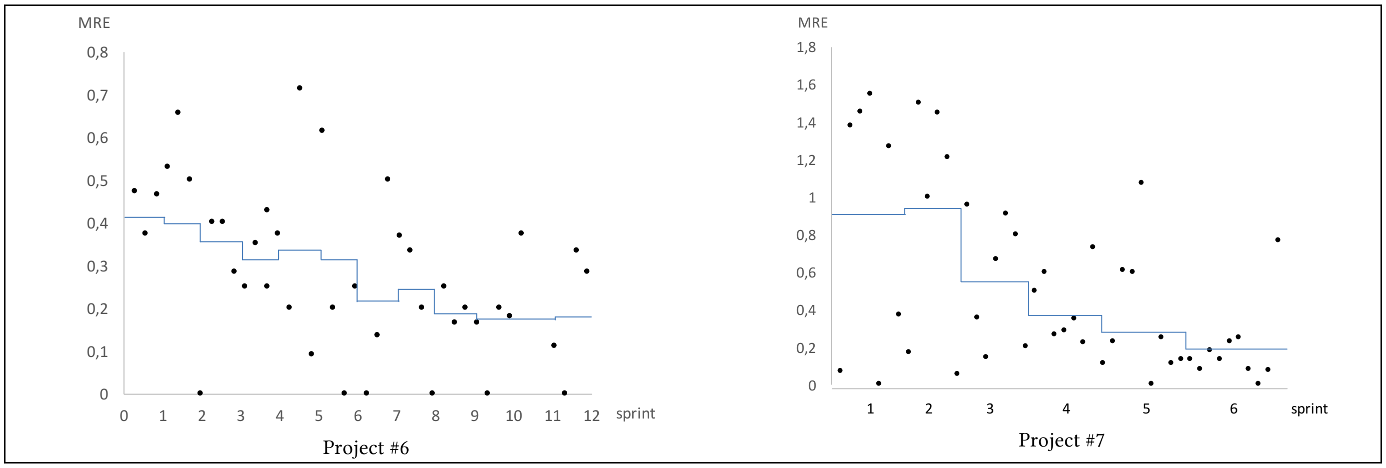

Figure 3: Estimation Accuracy (Q1)- Project using the Agile Experience Factory from the beginning

Table 1: Descriptive Statistics on User Stories (US) Effort.

Prj	#Dev.	Duration (weeks)	#US	Total effort (hours)	Actual effort per story (hours)			MRE		
					Mean	Max	Min	MMRE	Max	Min
1	6	8	25	863	34.5	120	4	52%	96%	0.00
2	8	9	75	285	3.8	56	1	41%	100%	0.00
3	7	14	153	903	5.9	60	0.25	66%	900%	0.00
4	7	150	129	14191	109.2	423	2.75	70%	357%	0.06
5	5	12(a) 25(b)	139	403	2.9	15	0.33	43%(a) 18%(b)	430%(a) 143%(b)	0%(a) 0%(b)
6	4	23	41	648	15.80	75	2	28%	71%	0%
7	5	12	46	719	15.63	137	4	46%	160%	0%

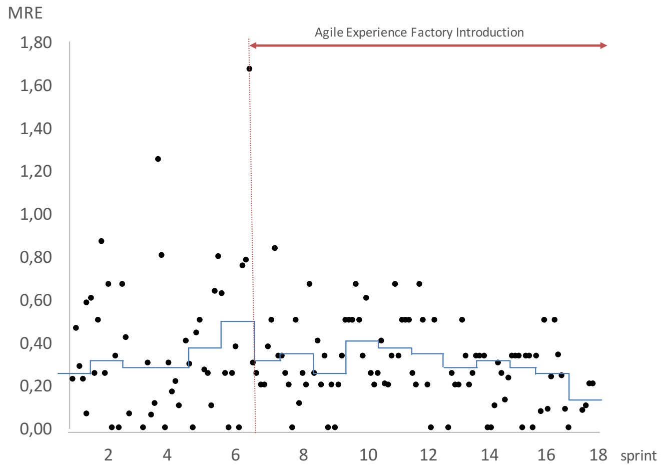

Figure 4: Estimation Accuracy (Q1) – Project #5

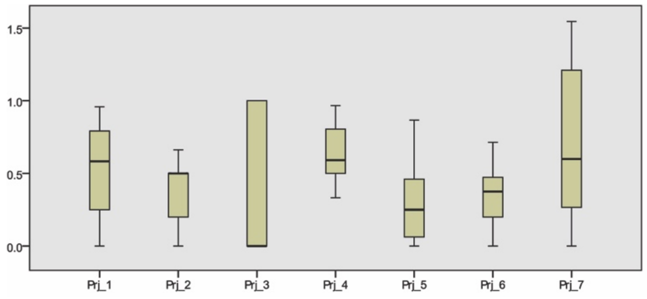

Figure 5:Estimation Accuracy (Box Plot)

Table 2: Q2 Results - Cronbach's α measure

	Usefulness	Underst.	Easiness	Self-Efficacy
Cronbach's α	0.87	0.92	0.89	0.85

Table 3: Q2 Results – Perceived Applicability

	Usefulness	Underst.	Easiness	Self-Efficacy
Mean	3.89	4.78	3.74	3.89
Median	4.00	5	3.67	4.00
Std. Dev.	0.52	0.44	0.72	0.55

5-point ordinal Likert scale: 1=strongly disagree, 2=disagree, 3=neither agree nor disagree, 4=agree, 5=strongly agree

5 DISCUSSION OF THE RESULTS

As confirmed by the obtained results, the Agile Experience Factory can be easily instantiated to effectively collect project-related measures. Some previous studies that attempted to collect data to support estimation in agile processes confirmed that functional size measures are not applicable and do not help developers to increase estimation accuracy [18] [19]. However, the results for the collection of effort per user story provided a very high ROIT, mainly because of the easiness of the collection. Compared to functional measures, in our case the process only requires storing the number of hours, without the need to make any calculations based on other parameters such as those required for functional measures. The actual and estimated times can be collected within a very short period of time, and no extra effort is needed to store them in the dataset. The only relatively small effort needed to use the data is that related to the search phase, when developers need to search the online spreadsheet for similar stories. However, probably due to the limited number of user stories, it took our developers only 1.5 minutes on average to search for previous similar stories, make the estimate, and store the estimated effort value for the new story to be developed.

Results show that the approach is highly beneficial in Scrum. In many Scrum teams, there is a tendency of not to use the retrospectives properly and this approach helps teams to reason on their project results and discussing about the metrics collected and on project results, even in case of useless measures (e.g. negative ROIT).

The analyzed projects ranged from small to medium size, with the developers working on the case study being mainly master and PhD students who worked part-time (from 8 to 20 hours per week) and had limited development skills.

Projects developed by industry practitioners might provide a different ROIT. However, considering the limited effort needed to collect the measure, we suggest that companies that will instantiate the factory should start with the collection of this measure.

6 THREATS TO VALIDITY

Concerning internal validity, the subjects were trained in the use of agile methodologies before the study started. Moreover, they were sufficiently motivated because the application of the case study in the effort estimation domain could have led to better personal performance. A standardized process was used for all the projects, which assured that experimenter expectancies did not influence the participants.

As for external validity, the web applications developed during the study were real applications, deployed in real environments, with their Minimum Viable Product (MVP)[38] defined by an entrepreneur. Moreover, the participants were trained well enough to be considered junior and senior developers.

Regarding conclusion validity, the questionnaires were checked by an expert on empirical studies. Although some of our results were not statistically significant, they indicate trends that can be validated in future studies.

Regarding construct validity, we followed the Goal Question Metric (GQM) methodology [10] define the goal, the questions, and the metrics. The goal was refined into clearly defined metrics to avoid misunderstandings, and the tasks were designed to avoid threats due to mono-operation bias. Each question was reviewed by an expert in empirical software engineering. The subjects were not aware of the hypotheses to be tested or the measures to be taken

7 CONCLUSIONS

In this paper, we proposed the Agile Experience Factory, an implementation/instantiation of the Experience Factory [7] aimed at collecting only those measures that provide a good Return on Invested Time (ROIT) and at learning from previous historical data.

In order to prove the applicability of the proposed approach, we ran a multiple-case study where we analyzed seven agile projects developed under similar conditions. In four projects, we tracked project performance without introducing the factory; in one project, we introduced the factory after one third of the development time; and in two projects, we instantiated the factory from the beginning. We instantiated the factory by collecting data on the actual and the estimated effort so as to understand whether we could improve effort accuracy simply by keeping track of this simple information.

Our experience with the case study on the Agile Experience Factory suggests that:

- It is possible to collect historical data on agile processes without affecting the speed of the development.
- An Agile Experience Factory can be effectively instantiated in an agile process, supporting developers in easily understanding which measure to keep or discard based on its Return on Invested Time (ROIT).
- The factory can help to increase the speed of a project, reducing the probability of wrong decisions caused by human errors.

However, although we tried to minimize threats to validity of the case study, results cannot be generalized.

Future work will include the application of the factory to different processes and the collection of different measures in order to validate the concept in different contexts in different industrial settings. Moreover, we aim at defining a more structured concept for collecting experiences and categorizing them.

REFERENCES

[1] P. Diebold, "How to Configure SE Development Processes Context-Specifically?," International Conference on Product Focused Software Process Improvement (PROFES), Paphos (Cyprus) 2013.

[2] P. Diebold, L. Dieudonné and D. Taibi, "Process Configuration Framework Tool," in 39th Euromicro Conference on Software Engineering and Advanced Applications, Verona, 2014.

[3] D. Taibi , L. Dieudonné and C. Plociennik, "A Classification Schema for Development Technologies," in ICSEA - International Conference on Software Engineering and Advances, Nice, 2014.

[4] L. Dieudonné, V. Lenarduzzi, C. Plociennik and D. Taibi, "Towards a Classification Schema for Development Technologies: an Empirical Study in the Avionic Domain." International Journal On Advances in Software, vol. 1&2, no. 8, pp. 125-135, 2015.

[5] V. Basili and H.-D. Rombach, "Towards A Comprehensive Framework for Reuse: A Reuse-Enabling Software Evolution Environment," in 19th Annual Software Engineering Workshop, Maryland, USA, 1988.

[6] V. Basili and H.-D. Rombach, "Support for Comprehensive Reuse," Software Engineering Journal, no. 6, pp. 303-316, 1991.

[7] V. Basili, G. Caldiera and H. Rombach, "Experience Factory," in Encyclopedia of Software Engineering, New York, John Wiley & Sons, 1994, pp. 469-476.

[8] P. D. a. T. Zehler, "The agile practices impact model: idea, concept, and application scenario," ICSSP - International Conference on Software and System Processes , New York, 2015.

[9] V. Basili, "Quantitative Evaluation of Software Engineering Methodology," in First Pan Pacific Computer Conference, Melbourne, 1985.

[10] G. Caldiera, H. D. Rombach and V. Basili, "Goal Question Metric Approach," in Encyclopedia of Software Engineering, NY, John Wiley & Sons, Inc., 1994, pp. 528-532.

[11] F. D. V. Venkatesh, "A theoretical extension of the technology acceptance model: Four longitudinal field studies," Management Science, vol. 46, no. 2, pp. 186-204, 2000.

[12] P. Runeson and M. Höst, "Guidlines for conducting and reporting case study research in software engineering," Empirical Software Engineering, no. 14, pp. 131-164, 2009.

[13] P. Diebold, C. Lampasona and D. Taibi, "Moonlighting Scrum: An Agile Method for Distributed Teams with Part-Time Developers Working during Non-Overlapping Hours," in ICSEA - International Conference on Software Engineering and Advances, Venice, 2013.

[14] V. Lenarduzzi and D. Taibi, "Can Functional Size Measures Improve Effort Estimation in SCRUM?," in ICSEA - International Conference on Software Engineering and Advances, Nice, 2014.

[15] V. Lenarduzzi, I. Lunesu, M. Matta and D. Taibi, "Functional Size Measures and Effort Estimation in Agile Development: a Replicated Study," in XP2015, Helsinki, 2015.

[16] A. Jedlitschka, "An Empirical Model of Software Managers' Information Needs for Software Engineering Technology Selection", Stuttgart: Fraunhofer IRB Verlag, 2009.

[17] G. Melnik and F. Maurer, "Direct verbal communication as a catalyst of agile knowledge sharing," Agile Development Conference, 2004.

[18] T. Chau, F. Maurer: "Knowledge Sharing in Agile Software Teams" Logic versus approximation LNCS 3075 , 2004

[19] T. Chau, F. Maurer, G. Melnik, "Knowledge Sharing: Agile Methods vs. Tayloristic Methods," WETICE, 2003

[20] T. Chau, F. Maurer: "Tool Support for Inter-team Learning in Agile Software Organizations" International Workshop, LSO 2004.

[21] M. Jorgensen, B. Boehm, and S. Rifkin. "Software development effort estimation: Formal models or expert judgment?." IEEE software 26.2 (2009): 14-19.

[22] E. Coelho, A. Basu. "Effort estimation in agile software development using story points." International Journal of Applied Information Systems (IJAIS),3(7).

[23] T. Menzies, Y. Yang, G. Mathew, B.. Boehm, J. Hihn. "Negative results for software effort estimation." Empirical Software Engineering, 1-26.

[24] R. Popli, C. Naresh "Cost and effort estimation in agile software development." ICROIT, 2014 International Conference on. IEEE, 2014.

[25] S. Ziauddin, S. Kamal, S. Zia. "An effort estimation model for agile software development." Advances in Computer Science and Its Applications (ACSA) 314 2.1 (2012): 314-324.

[26] E. Mendes. "Applying a Knowledge Management Technique to Improve Risk Assessment and Effort Estimation of Healthcare Software Projects." ICST. Heidelberg, 2013.

[27] T. Chau, T. Maurer, and G. Melnik. "Knowledge sharing: Agile methods vs. tayloristic methods." Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on. IEEE, 2003.

[28] M. Ajmal, K.U. Koskinen. "Knowledge transfer in project-based organizations: an organizational culture perspective." Project Management

[29] B.Tanveer, L. Guzmán, and U. M. Engel. "Understanding and improving effort estimation in agile software development: An industrial case study."Proceedings of the International Workshop on Software and Systems Process. ACM, 2016.

[30] M. Usman, E. 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 (pp. 82-91). ACM.

[31] R. Yin. "Case Study Research: Design and Methods" SAGE publishing (2014) ISBN 9781452242569

[32] V.Lenarduzzi, I. Lunesu, M. Matta and D. Taibi."Functional size measures and effort estimation in agile development: A replicated study" XP2015 Agile Processes in Software Engineering and Extreme Programming pp 105-116, 2015.

[33] V. Lenarduzzi, S. Morasca and D. Taibi. "Estimating software development effort based on phases". Euromicro Conference on Software Engineering and Advanced Applications, pp. 305-308, 2014.

[34] L. Lavazza, S. Morasca, D. Taibi and D. Tosi. "Applying SCRUM in an OSS development process: An empirical evaluation". XP2010 Agile Processes in Software Engineering and Extreme Programming pp 147-159, 2010.

[35] L. Lavazza, S. Morasca, D. Taibi and D. Tosi. "On the definition of dynamic software measures". ESEM2012, International Symposium on Empirical Software Engineering and Measurement, pp. 39-48, 2012.

[36] D. Taibi, L. Lavazza, S. Morasca, and D. Tosi. "An empirical investigation of perceived reliability of open source Java programs" SAC '12, Annual ACM Symposium on Applied Computing, 2012.

[37] S. Morasca, D. Taibi and D. Tosi. "Towards certifying the testing process of open-source software: New challenges or old methodologies?". FLOSS '09. ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development, pp. 25-30, 2009.

[38] D. Taibi and Lenarduzzi, V. , "MVP explained: A Systematic Mapping on the Definition of Minimum Viable Product", in SEAA2016 42th Euromicro Conference on Software Engineering and Advanced Applications 2016, Cyprus, 2016.