©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

# Towards the Statistical Construction of Hybrid Development Methods

**Paolo Tell*[1] | Jil Klünder[2] | Steffen Küpper[3] | David Raffo[4] | Stephen G. MacDonell[5] | Jürgen Münch[6] | Dietmar Pfahl[7] | Oliver Linssen[8] | Marco Kuhrmann[9]**

[1]IT University Copenhagen, Denmark,
Email: pate@itu.dk
[2]Leibniz University Hannover, Germany,
Email: jil.kluender@inf.uni-hannover.de
[3]Clausthal University of Technology, Germany
Email: steffen.kuepper@tu-clausthal.de
[4]Portland State University, USA,
Email: raffod@pdx.edu
[5]Auckland University of Technology, New Zealand,
Email: smacdone@aut.ac.nz
[6]Reutlingen University, Germany,
Email:
juergen.muench@reutlingen-university.de
[7]University of Tartu, Estonia,
Email: dietmar.pfahl@ut.ee
[8]FOM Hochschule für Oekonomie & Management, Germany,
Email: oliver.linssen@fom.de
[9]University of Passau, Germany
Email: kuhrmann@acm.org

**Correspondence**
*Corresponding author: Email: pate@itu.dk

## Abstract

Hardly any software development process is used as prescribed by authors or standards. Regardless of company size or industry sector, a majority of project teams and companies use hybrid development methods (short: hybrid methods) that combine different development methods and practices. Even though such hybrid methods are highly individualized, a common understanding of how to systematically construct synergetic practices is missing. In this article, we make a first step towards a statistical construction procedure for hybrid methods. Grounded in 1,467 data points from a large-scale practitioner survey, we study the question: What are hybrid methods made of and how can they be systematically constructed? Our findings show that only eight methods and few practices build the core of modern software development. Using an 85% agreement level in the participants' selections, we provide examples illustrating how hybrid methods can be characterized by the practices they are made of. Furthermore, using this characterization, we develop an initial construction procedure, which allows for defining a method frame and enriching it incrementally to devise a hybrid method using ranked sets of practice.

**KEYWORDS:**
software development, software process, hybrid methods, survey research

## 1 | INTRODUCTION

Today, companies often use highly individualized processes to run projects, often by integrating agile methods in their processes. For instance, Dikert et al.[1] found choosing and customizing an agile model to be an important success factor, and that agility in general changed the way software is developed. Dingsøyr et al.[2] reflect on a decade of agile methodologies and there is no denial that agile methods have become an important asset in many companies' process portfolios[3,4,5,6]. However, agile methods are not implemented as prescribed by authors or standards[7,8], and in 2011, West et al.[9] coined the term *"Water-Scrum-Fall"* to describe a pattern which they claimed most companies implement for their software projects.

In previous research[10,11], we could confirm West's claim. In addition, independently conducted research[12] and a number of country-specific[4,13], and industry-hosted studies[14] provide evidence on the use of *hybrid development methods*[10,15] (short: hybrid methods). Although agile methods have been an important research topic, they have also stimulated increasing diversity in software and system development, little information is available about the nature of hybrid development methods, what they look like, and how to devise them.

**Problem Statement.** Modern software and system development does not follow any blueprint. A variety of different frameworks, methods, and practices are used in practice; according to a study by Klünder et al.[16], 78.5% of practitioners evolve their processes over time to improve, for instance, different product quality attributes and to keep flexibility regarding the ability to react to change. However, an understanding of what a

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

hybrid development method is composed of is missing, e.g., which combinations of frameworks, methods, and practices for software and system development help practitioners implement a process environment that provides the company and the management with a stable framework while providing developers with the demanded flexibility[6,11].

**Objective.** The work presented in this paper aims to lay the foundation for understanding *hybrid development methods* and to develop adaptable construction procedures that help devise such methods grounded in evidence. The objective of our research is to *understand which frameworks, methods, and practices are used to realize hybrid methods in practice and to provide an evidence-based characterization of such methods*.

**Contribution.** Based on a large-scale international online survey, we analyze 1,467 data points that provide information about the combined use of 60 frameworks, methods, and practices. Our findings indicate that using hybrid development methods *is* the norm in modern software and system development, and that using hybrid methods happens in companies of all sizes and across all industry sectors. We identify eight base methods providing the basis for devising hybrid methods, and we statistically compute sets of practices used to embody the base methods. We contribute a statistical process that helps computing hybrid methods (including process variants) to provide advice to practitioners what (not) to include in their process portfolio.

**Context.** The research presented in this paper emerges from the HELENA (Hybrid dEveLopmENt Approaches in software systems development, online: https://helenastudy.wordpress.com) study, which is a large-scale international online survey in which a team of 75 researchers and practitioners from 25 countries collected data world wide. The study was implemented in two stages (Figure 2) of which the first stage was a public trail executed in 2016 in Europe[15,10]. All data and complementing materials of the second stage are available online[17].

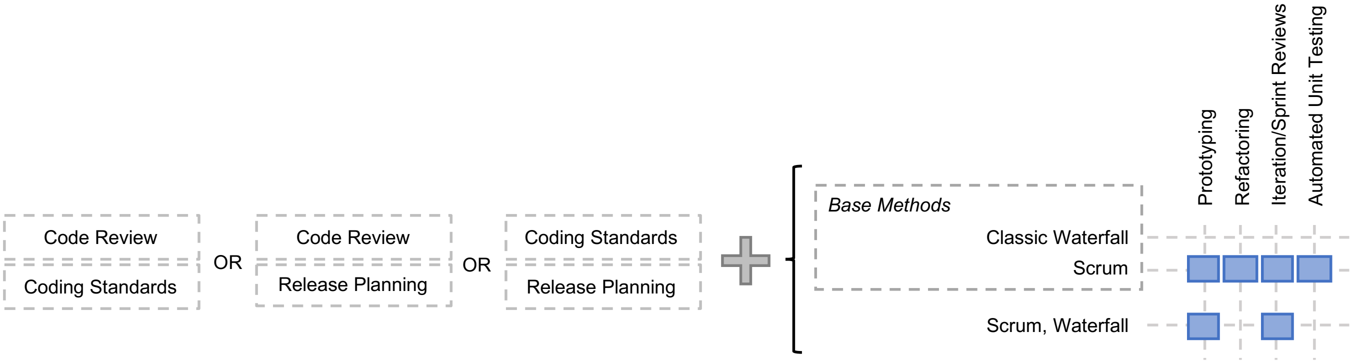


**FIGURE 1** An example of a constructed hybrid method as shown in Tell et al.[18] that illustrates West's *Water-Scrum-Fall*[9]. The figure shows the base methods, the statistically computed core practices and the (*Scrum–Waterfall*) method, complemented with the different practices that shape the hybrid method, e.g., *Prototyping*.

This article is an extended version of the previously published ICSSP conference paper[18] in which we studied the statistical construction of hybrid methods. For instance, Figure 1 shows how such a construction works: hybrid methods are computed based on statistically constructed base methods, method combinations, and core practices according to level of agreement within the dataset. These hybrid methods include a method combination as framework to host different practices. Due to the availability of different core practice clusters and several extra practices reported relevant by the HELENA study participants, in our previous paper[18], we constructed hybrid methods and variants of such methods. The extension of our previously published paper is thus shaped by (1) an improved discussion of our work in the context related contributions and, moreover, (2) an extended analysis of the data. Specifically, we extended our discussion on how to utilize the analysis procedure documented in the conference paper[18] and derived an initial proposal for a *statistical construction procedure for hybrid development methods*. A new research question covers the necessary extra analyses, which aim to better characterize combinations of frameworks, methods, and practices forming hybrid development methods.

**Outline.** The paper is organized as follows: Section 2 presents related work. In Section 3, we present the research design. The results are presented in Section 4 and discussed in Section 5. The paper is concluded in Section 6.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

## 2 | RELATED WORK

The use of software development processes has been studied since the 1970s, when the first ideas to structure software development appeared[19,20]. Since then, a growing number of approaches emerged, ranging from traditional and rather sequential models, to iterative and agile models and a number of survey studies have sought to investigate the state of practice by focusing on software development methods. For instance, the "State of Agile" survey[14] annually collects data on the use of agile methods. The "Swiss Agile Study"[13] and the "Status Quo Agile" study[21] collect data in certain intervals aiming at observing the use of agile methods in Switzerland and Germany. Garousi et al.[4] provide an overview of the use of agile methods in Turkey. Tripp and Armstrong[22] investigated the "most popular" agile methods and found XP, Scrum, Dynamic Systems Development Method (DSDM), Crystal, Feature Driven Development (FDD), and Lean Software Development among the top methods used. These studies, however, explicitly focus on agile methods and cover only some of the traditional methods. Dingsøyr et al.[2] provided an overview of "a decade of agile software development", and motivate research towards a rigorous theoretical framework of agile software development, specifically, on methods of relevance for industry. However, Theocharis et al.[11] provided evidence that this focus on agile is too narrow as, for instance, numerous companies and project teams remain skeptical and do not consider agile methods as the "Silver Bullet"[23,6,24,25]. For example, Cusumano et al.[26] surveyed 104 projects and found many using and combining different development approaches. In an analysis of 12,000 projects, Jones[27] found that both specific design methods and programming languages can lead to successful or troubled project outcome. Neill and Laplante[28] found that approximately 35% of developers used the classical Waterfall model. However, projects also used incremental approaches, even within particular lifecycle phases.

It is a matter of fact that companies develop a heterogeneous process portfolio comprised of a variety of traditional and agile methods and practices. For instance, Cockburn[29] describes a framework to choose appropriate methods to address the needs of projects. Boehm and Turner[30] aimed to overcome situation-specific shortcomings of agile and plan-driven development by defining five factors that describe a project environment and help determine a balanced method. Different complementary research streams were developed to address the required process variability and adaptability demands. For instance, Clarke and O'Connor[31] and Kalus and Kuhrmann[32] provide collections of situational factors or tailoring criteria to support process tailoring. Another research stream is focused on software process lines[33,34], which aim to provide a comprehensive framework that allows for defining composite development methods, which comply with a standard/reference process model. Further initiatives to provide a structured approach towards modeling simple and complex heterogenous development methods are, among other things, reflected in initiatives such as SEMAT and Essence[35,36,37], the standard ISO/IEC 24744:2007[38], and numerous activities in the context of the SPEM software process engineering metamodel[39,40]. All these initiatives aim to bring more flexibility to processes and to help companies devise context-specific processes. Solinski and Petersen[41] aimed to characterize such combined processes and found Scrum and XP to be the most commonly adopted methods, with Waterfall/XP and Scrum/XP as the most common combinations. For such combined processes, West et al.[9] coined the term *"Water-Scrum-Fall"*, and different studies[11,42,12] provide evidence that the use of hybrid methods has become the norm. In 2017, we generalized this concept, defining the term "*hybrid development methods*" as "*any combination of agile and traditional (plan-driven or rich) approaches that an organizational unit adopts and customizes to its own context needs*"[10].

Available studies thus show a situation in which traditional and agile approaches coexist and form the majority of practically used hybrid methods. In contrast, current literature on software processes and their application in practice leaves researchers and practitioners with an increasing amount of research focusing on agile methods only. Traditional models are vanishing from researchers' focus. They only play a role in process modeling, in domains with special requirements (e.g., regulations and norms), or in discussions regarding the reasons why certain companies do not use agile methods[6,22]. In our previously published papers[15,10], we initially studied the state of practice in using different frameworks, methods, and practices in combination and derived process clusters that form hybrid development methods. Klünder et al.[16] studied the development of hybrid methods and found an evolutionary approach to be the common way to devise such methods, followed by planning a method as part of software process improvement programs.

The article at hand extends the available research by investigating the characteristics of hybrid development methods with a specific focus on the components of hybrid methods. We analyze combinations of frameworks, methods, and practices statistically to find such combinations that have a high level of agreement among the study participants and, thus, can be considered common sense about the basic structure of hybrid methods. Beyond the plain analysis, we also make a first step towards constructing hybrid methods by describing a statistical procedure that helps in computing hybrid methods and their variants from data.

## 3 | RESEARCH DESIGN

We present the research design including the research questions, information about the survey instrument, and the different procedures regarding data collection, analysis and the validity. The overall research design is outlined in Figure 2, the individual steps are described in the following

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

paragraphs. Figure 2 also highlights the relation to the initial publication[10], which was published based on the results of the first stage of HELENA and that lays the foundation for our original ICSSP conference paper[18] and the article at hand.

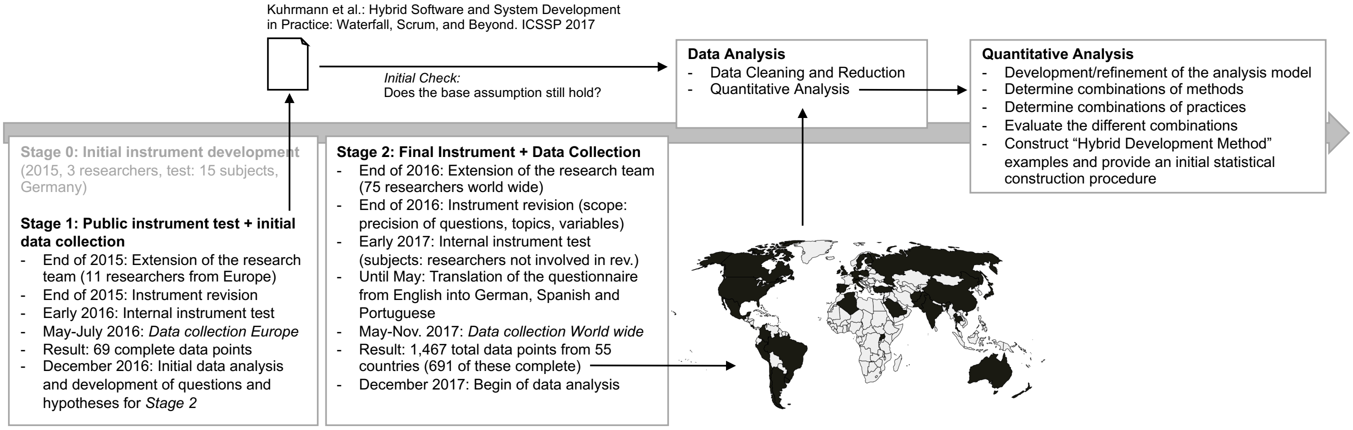

Kuhrmann et al.: Hybrid Software and System Development in Practice: Waterfall, Scrum, and Beyond. ICSSP 2017
Initial Check:
Does the base assumption still hold?
Data Analysis
- Data Cleaning and Reduction
- Quantitative Analysis
Quantitative Analysis
- Development/refinement of the analysis model
- Determine combinations of methods
- Determine combinations of practices
- Evaluate the different combinations
- Construct "Hybrid Development Method" examples and provide an initial statistical construction procedure
Stage 0: Initial instrument development
(2015, 3 researchers, test: 15 subjects, Germany)
Stage 1: Public instrument test + initial data collection
- End of 2015: Extension of the research team (11 researchers from Europe)
- End of 2015: Instrument revision
- Early 2016: Internal instrument test
- May-July 2016: Data collection Europe
- Result: 69 complete data points
- December 2016: Initial data analysis and development of questions and hypotheses for Stage 2
Stage 2: Final Instrument + Data Collection
- End of 2016: Extension of the research team (75 researchers world wide)
- End of 2016: Instrument revision (scope: precision of questions, topics, variables)
- Early 2017: Internal instrument test (subjects: researchers not involved in rev.)
- Until May: Translation of the questionnaire from English into German, Spanish and Portuguese
- May-Nov. 2017: Data collection World wide
- Result: 1,467 total data points from 55 countries (691 of these complete)
- December 2017: Begin of data analysis

**FIGURE 2** Overview of the research method applied including this study's position in the overall HELENA project and its previously published outcomes. The figure also includes a world map that highlights all 55 countries that contributed to the HELENA dataset.

## 3.1 | Research Objective and Research Questions

The overall objective of the research presented in this paper is to understand which frameworks, methods, and practices are used to realize hybrid development methods in practice, to provide an evidence-based characterization of such methods, and to develop a *construction procedure*, which is grounded in evidence and allows for systematically constructing hybrid development methods. For this, we study the following research questions:

**RQ1:** *Which frameworks and methods form the basis for devising hybrid development methods?* This question sets the scene by analyzing the more comprehensive frameworks and methods that form the basis for hybrid methods and bind the different (smaller) practices together. This research question is motivated by a finding from our previous study[10] that process clusters are formed around "centers". The first step is thus to identify such centers. As the HELENA study contains a flag that indicates if a specific set of frameworks, methods and practices is intentionally used in combination, the analysis is performed twice: once for the entire dataset and once for the subset of data for which the study participants explicitly stated to combine the different processes.

**RQ2:** *Which practices are used to embody method combinations for devising hybrid development methods?* Having identified the base methods and the method combinations providing the frame for a hybrid method, we analyze the data for recurring practices used to embody the identified base methods and method combinations. That is, we aim to identify specific combinations of frameworks, methods *and* practices that, together, form hybrid development methods. Again, the investigation is performed twice for the entire dataset and the subset of participants that explicitly combine processes.

**RQ3:** *Which process variants are more promising?* The results from Tell et al.[18] showcased a significant variability in the combinations of practices within each hybrid method leading to additional questions. For instance, is it possible to identify *preferred process variances* within the dataset? Therefore, we aim to further leverage the construction process[18] to qualify process variability and to improve our understanding of promising process variants, i.e., practitioners' *preferred* framework, method, and practice combinations.

**RQ4:** *How can hybrid development methods be characterized?* In Tell et al.[18], we drafted an initial procedure with which we utilize the analysis procedures applied to the HELENA dataset to statistically construct hybrid methods. Including the findings from RQ 3, this research question aims to improve that procedure to help characterize hybrid methods by defining core practices that, together with the base methods and method combinations, provide a means to devise hybrid methods. Hence, we aim to statistically define hybrid methods, to define a hybrid method and its variant space to provide a ranked list of suitable hybrid methods and, eventually, to help practitioners decide what to (not) include into their process portfolio.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

## 3.2 | Instrument Development and Data Collection

Data was collected using the survey method[43]. We designed an online questionnaire to solicit data from practitioners about the development approaches they use in their projects. The *unit of analysis* was either a project (ongoing or finished) or a software product.

### 3.2.1 | Instrument Development and Structure

The survey instrument was developed and refined in several iterations, which are illustrated in Figure 2. Finally, the research team included 75 researchers from all over the world. The questionnaire was made available in English, German, Spanish, and Portuguese and consisted of five parts (with number of questions[1]): Demographics (10), Process Use (13), Process Use and Standards (5), Experiences (2), and Closing (8). In total, the questionnaire comprised up to 38 questions, depending on previously given answers. A complete overview of the questionnaire structure, the questions, variables and datatypes, and the conditional paths through the questionnaire is available from the accompanying research kit[17], which also includes the raw questionnaire designs and their translations into the different target languages for independent replications, and the final raw dataset.

### 3.2.2 | Data Collection

Data were collected from May to November 2017 following a *convenience sampling strategy*[43]. The survey was promoted through personal contacts of the participating researchers, posters at conferences, as well as posts to mailing lists, social media channels (Twitter, Xing, LinkedIn), professional networks, and websites (ResearchGate and researchers' individual and/or institution home pages).

## 3.3 | Data Analysis Procedures

As outlined in Figure 2, the data analysis consisted of multiple parts, which are described in detail in this section.

### 3.3.1 | Data Cleaning and Data Reduction

The first step was the preparation of the data. We opted for the full dataset of the second stage of the HELENA study[17], which consists of 1,467 data points. As many questions were optional and participants had the opportunity to skip mandatory questions, we first analyzed the data for *NA* (i.e., "not available") and *-9* values. While *NA* values indicate that participants did not provide information for an optional question, *-9* indicates that participants skipped a mandatory question. Depending on the actual question, *-9* values were either transformed into *NA* values or the respective data point was excluded from further analysis because we considered the question not completely answered. Finally, in the question about company size (question D001[17]), we combined the categories *Micro* and *Small* into a new category *Micro and Small (1–50 employees)* that resulted in an almost even distribution among all company sizes.

### 3.3.2 | Checking the Base Assumptions

In this study, we are interested in the particular process combinations used in industry. Our base assumption is that frameworks, methods, and practices are combined in practice as claimed by West et al.[9]. For this, in our previous studies[10,15], we quantitatively analyzed the initial data using a set of hypotheses. As the first step in the quantitative data analysis, we tested the two hypotheses shown in Table 1 using Pearson's $\chi^2$ test at a significance level of 0.05.

**TABLE 1** Null hypotheses used to check the base assumption that combinations are common practice.

| Hypotheses | | Question/Variable Assignment to Hypotheses |
|---|---|---|
| $H1_0$ | The use of hybrid methods does not depend on the company size. | Combination (PU04), Company size (D001) |
| $H2_0$ | The use of hybrid methods does not depend on the industry target domain. | Combination (PU04), Industry sector (D005) |

[1] An important aspect to note is that no definition was provided for any given item, be this a method, a practice, or a technical concept in general. This was done to avoid any bias that might have been introduced from exposing participants to our knowledge, informing them on the concepts of interest, or leading them towards specific attitudes when answering the survey questions.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

While H1 was directly tested using Pearson's $\chi^2$ test, testing H2 required a different procedure as participants were able to provide more than one industry sector as targets for the question D005[17]. Therefore, a Pearson's $\chi^2$ test was evaluated for all industry sectors. For each industry sector, we tested the share of participants stating that they (do not) combine the different frameworks, methods, and practices and compared those with all the other industry sectors. As the number of data points per industry sector influences the p-value, we used all selections of the respective industry sectors as sample size for the $\chi^2$ tests. Finally, because we tested a single hypothesis using multiple tests, we used a *Bonferroni correction* to adjust the significance level by dividing the given significance level of 0.05 by the number of tests (the Bonferroni correction is used when several statistical tests are performed simultaneously, which requires an adjustment of the $\alpha$ value[44]). Including the option "Other" in question D005, we provided 20 industry sectors to choose from, i.e., the corrected significance level is $p_{B\_cor} \leq \frac{0.05}{20} = 0.0025$.

### 3.3.3 | Quantitative Analysis for Process Combinations

To derive process combinations from the data, we analyzed the (combined) occurrence of frameworks, methods, and practices in the dataset. For this analysis, we used the questions PU09 (frameworks and methods), PU10 (practices), and PU04 (combined process use). Our definition of *methods* and *practices* is grounded in the definitions for these concepts as provided by Diebold and Zehler[45]. They define an (agile) method as a process combination addressing the whole software lifecycle, while an (agile) practice is defined as an established instruction with a specific focus. We adopted these definitions such that they are applicable also to traditional and generic methods (frameworks) and practices. To structure the analysis, we defined the *analysis model* shown in Figure 3.

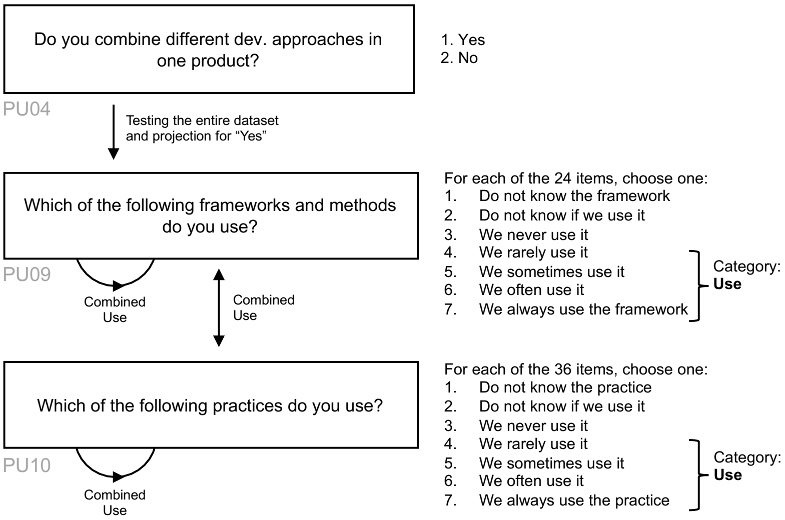

Do you combine different dev. approaches in one product?
1. Yes
2. No
PU04
Testing the entire dataset and projection for "Yes"
Which of the following frameworks and methods do you use?
For each of the 24 items, choose one:
1. Do not know the framework
2. Do not know if we use it
3. We never use it
4. We rarely use it
5. We sometimes use it
6. We often use it
7. We always use the framework
Category: Use
PU09
Combined Use
Combined Use
Which of the following practices do you use?
For each of the 36 items, choose one:
1. Do not know the practice
2. Do not know if we use it
3. We never use it
4. We rarely use it
5. We sometimes use it
6. We often use it
7. We always use the practice
Category: Use
PU10
Combined Use

**FIGURE 3** Overview of the analysis model used in this study.

The analysis was performed in multiple steps and each step was performed twice: (i) on the entire dataset and (ii) on a subset created from a filter using the participants' selection of question PU04, i.e., the selection if participants combine different methods intentionally or not. Specifically, the following main analysis steps using the R-package *apriori* (online: https://www.rdocumentation.org/packages/arules/versions/1.6-2/topics/apriori) that, among other features, allows for setting recurrence thresholds and minimum/maximum set sizes, were performed:

**Methods:** First, the combined use of the different frameworks and methods, e.g., *Waterfall*, *DevOps*, and *Kanban*, was analyzed, and a Top-10-like list of methods and method combinations was computed. The combinations were computed using a recurrence threshold of 35%, i.e., we included methods and combinations that were selected by at least 35% of the participants. The recurrence threshold was set to 35% as it identifies a minimal group of three frameworks and methods in the entire dataset, and a minimal group of four in the projected dataset generated through PU04="Yes".

**Practices:** Similarly, we analyzed the practices, e.g., *Coding Standards*, *Code Reviews*, and *Release Planning*. Different to the analysis of the frameworks and methods, we used an 85% recurrence threshold as this threshold provides a minimal group of two practices in the entire dataset as well as in the projected dataset generated through PU04="Yes".

ⓒWiley. PREPRINT. This is the author's version of the work. Not for redistribution.

**Process Variants:** Finally, one exemplary selected hybrid method (i.e., *Scrum*, *Iterative Development*, and *Lean* in the filtered dataset generated through PU04="Yes") was further investigated to qualify the process variants. This particular hybrid was selected as a demonstrator for the analysis procedure as having the highest number of process variants. To this end, for each size of combinations, all combinations of practices reaching 85% recurrence threshold within the hybrid were sorted based on the recurrence threshold, and the set of practices sorted based on their first appearance index.

### 3.4 | Validity Procedures

To improve the validity and to mitigate risks, we implemented different measures focused around replicability and consistency as well as bias. First, our research is grounded in previously conducted studies. Notably, the key question of this study was derived from the outcomes of our previously conducted study[10]. An extended design team developed the survey instrument as described in[17]. The data analysis was performed by different teams, i.e., one team performed the hypothesis testing while another team focused on the quantitative analyses. Researchers not involved in the data analysis were tasked to provide the quality assurance.

Second, as one of the main goals of this study is to build a quantitative basis, we opted for the *convenience sampling strategy*[43] to collect the data by accepting the risk of losing full control in terms of sampling, response rate and so forth. This decision was made to collect as many data points as possible. To handle this risk, before analyzing the data, we implement rigorous data pre-processing including a consistency check of the data (see Section 3.3.1).

## 4 | RESULTS

In this section, we present the results of our study. The presentation of the results is structured following the research questions as provided in Section 3.1 and our analysis model as shown in Figure 3.

### 4.1 | Checking the Base Assumptions

As outlined in Section 3.3.2, our study is built on previously published studies[10,15] that found no evidence that the use of combined processes in practice depends on company size or industry sector. Therefore, we tested two hypotheses for which the results are presented in Table 2 (H1; from Kuhrmann et al.[10]) and in Table 3 (H2; according to Kuhrmann et al.[15]).

**TABLE 2** Result of testing $H1_0$: The use of hybrid methods does not depend on the company size.

| Id | Results | Decision | Kuhrmann et al.[10] |
|---|---|---|---|
| $H1_0$ | $\chi^2$ = 1.9972, df = 3, p $= 0.573$ | no support | no support |

Table 3 shows the ratios of participants that do not combine (*non-hybrid*, NH) and those that combine processes (*hybrid*, H) within an industry sector and for all remaining industry sectors. The table shows the individual test results, which, however, have to be considered in the context of the Bonferroni-corrected significance level of $p_{B\_cor} \leq 0.0025$ (Section 3.3.3). The results shown in Table 2 and in Table 3 support the findings from[10,15]. Notably, the results from Table 3—given the Bonferroni correction—show that no $\chi^2$ test is significant, which does not allow for concluding that the industry sector influences the use of hybrid methods. Hence, the results show that the combined use of different frameworks, methods, and practices, i.e., the use of hybrid methods, is a common practice in industry. The question for *"What do such combinations look like?"* has therefore to be considered of high relevance.

### 4.2 | Combined Use of Frameworks and Methods

The first step in the quantitative analysis is the investigation of the combined use of frameworks and methods. Of the 1,467 data points, 845 provide data on the use of frameworks and methods, i.e., answers to question PU09 (Figure 3). As shown in Figure 3, this multiple-choice question provided 24 items to choose from complemented with a free-text option. Of the 845 data points, 792 had multiple selections. The quantitive

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

**TABLE 3** Results for testing $H2_0$: The use of hybrid methods does not depend on the industry target domain (with corrected significance level $p_{B\_cor} \leq 0.0025$ for 20 industry target domain options; NH: not hybrid, i.e., PU04 = "No"; H: hybrid, i.e., PU04 = "Yes").

| Industry Sector | This Sector NH:H | Other Sectors NH:H | $\chi^2$ | p-value |
|---|---|---|---|---|
| Automotive Software and Systems | 14 : 66 | 163 : 515 | 1.37 | 0.24 |
| Aviation | 09 : 21 | 168 : 560 | 0.43 | 0.51 |
| Cloud Application and Services | 29 : 95 | 148 : 486 | 0.00 | 1.00 |
| Defense Systems | 02 : 26 | 175 : 555 | 3.38 | 0.07 |
| Energy | 07 : 30 | 170 : 551 | 0.21 | 0.65 |
| Financial Services | 34 : 149 | 143 : 432 | 2.73 | 0.10 |
| Games | 01 : 17 | 176 : 564 | 2.32 | 0.13 |
| Home Automation and Smart Buildings | 05 : 17 | 172 : 564 | 0.00 | 1.00 |
| Logistics and Transportation | 11 : 43 | 166 : 538 | 0.14 | 0.71 |
| Media and Entertainment | 06 : 25 | 171 : 556 | 0.10 | 0.75 |
| Medical Devices and Health Care | 13 : 61 | 164 : 520 | 1.20 | 0.27 |
| Mobile Applications | 19 : 105 | 158 : 476 | 4.82 | 0.03 |
| Other Embedded Systems and Services | 09 : 46 | 168 : 535 | 1.22 | 0.27 |
| Other Information Systems | 20 : 87 | 157 : 494 | 1.22 | 0.27 |
| Public Sector and Contracting | 21 : 72 | 156 : 509 | 0.00 | 0.95 |
| Robotics | 01 : 17 | 176 : 564 | 2.32 | 0.13 |
| Space Systems | 08 : 26 | 169 : 555 | 0.00 | 1.00 |
| Telecommunication | 07 : 38 | 170 : 543 | 1.19 | 0.27 |
| Web Applications and Services | 40 : 162 | 137 : 419 | 1.68 | 0.20 |
| Other | 27 : 63 | 150 : 518 | 2.12 | 0.15 |

**Finding 1:** The use of hybrid development methods has not shown any dependence with regards to either the company size (H1) or the industry sector (H2). Therefore, given the high p-value of the majority of the tests, the use of hybrid development methods can be considered state of practice across companies of all sizes and in all industry sectors.

analysis of the combined use of frameworks and methods is performed twice: once for all data points and again for those data points for which the study participants stated to use hybrid methods intentionally.

**Analysis of All Data Points.** Figure 4 (left) shows the resulting combinations using the 35% threshold for the combined process use in the entire (non-filtered) dataset. This threshold results in 17 groups of two or three combined frameworks and methods—there is no group with four or more elements with at least 35% agreement regarding the combined process use. *Scrum* is the most frequently selected method (674 participants), which is followed by *Iterative Development* (620) and *Kanban* (523). Extending the scope to framework and method combinations, a number of couples and *all* triplets include *Scrum*. Expected combinations are present, e.g., *(Scrum–Kanban–DevOps)*, which was stated by 309 participants, and the "Water-Scrum-Fall", i.e., *(Waterfall–Scrum)*, according to West et al.[9], which was mentioned by 380 participants. Please note that, in the following, we consistently present clusters of frameworks, methods and practices in the form *(*$Item_1$*–...–*$Item_n$*)* to make clear, which n frameworks, methods, and practices together form a cluster, i.e., a group that can be combined with further individual frameworks, methods, practices, and even other clusters.

**Analysis of Intentionally Combining Practitioners.** Applying the explicitly stated combinations of methods and practices, i.e. question PU04 (Figure 3) as a filter, i.e., re-running the analysis for only those participants that explicitly claimed to use the different frameworks, methods, and practices in combination, Figure 4 (right) results in 27 groups of two to four explicitly combined frameworks and methods, whereas there is no group with five elements or more having at least 35% agreement regarding combined process use. The combined frameworks and methods as shown

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

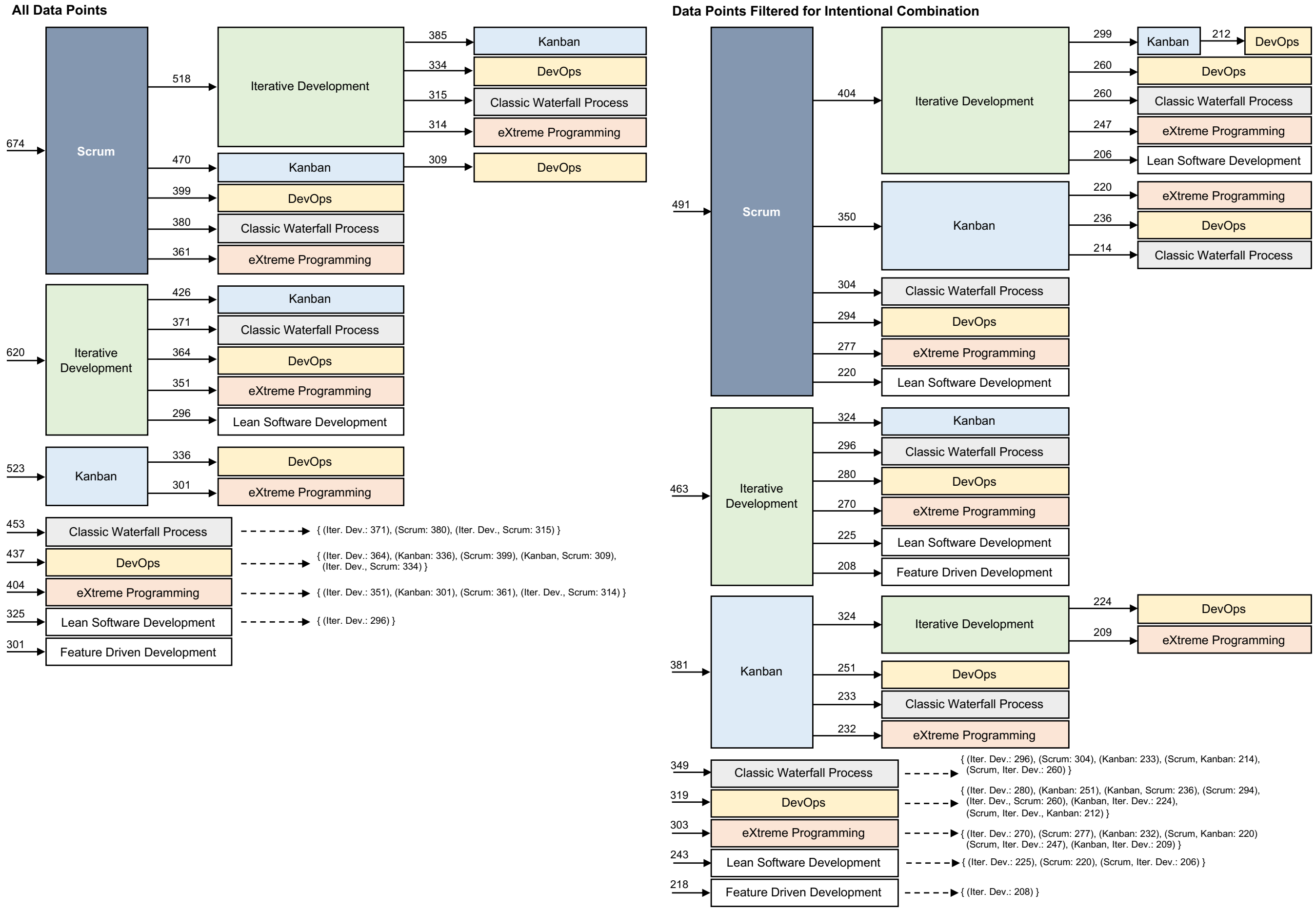

All Data Points
674
Scrum
518
Iterative Development
385
Kanban
334
DevOps
315
Classic Waterfall Process
314
eXtreme Programming
470
Kanban
309
DevOps
399
DevOps
380
Classic Waterfall Process
361
eXtreme Programming
620
Iterative Development
426
Kanban
371
Classic Waterfall Process
364
DevOps
351
eXtreme Programming
296
Lean Software Development
523
Kanban
336
DevOps
301
eXtreme Programming
453
Classic Waterfall Process
{ (Iter. Dev.: 371), (Scrum: 380), (Iter. Dev., Scrum: 315) }
437
DevOps
{ (Iter. Dev.: 364), (Kanban: 336), (Scrum: 399), (Kanban, Scrum: 309), (Iter. Dev., Scrum: 334) }
404
eXtreme Programming
{ (Iter. Dev.: 351), (Kanban: 301), (Scrum: 361), (Iter. Dev., Scrum: 314) }
325
Lean Software Development
{ (Iter. Dev.: 296) }
301
Feature Driven Development
Data Points Filtered for Intentional Combination
491
Scrum
404
Iterative Development
299
Kanban
212
DevOps
260
DevOps
260
Classic Waterfall Process
247
eXtreme Programming
206
Lean Software Development
350
Kanban
220
eXtreme Programming
236
DevOps
214
Classic Waterfall Process
304
Classic Waterfall Process
294
DevOps
277
eXtreme Programming
220
Lean Software Development
463
Iterative Development
324
Kanban
296
Classic Waterfall Process
280
DevOps
270
eXtreme Programming
225
Lean Software Development
208
Feature Driven Development
381
Kanban
324
Iterative Development
224
DevOps
209
eXtreme Programming
251
DevOps
233
Classic Waterfall Process
232
eXtreme Programming
349
Classic Waterfall Process
{ (Iter. Dev.: 296), (Scrum: 304), (Kanban: 233), (Scrum, Kanban: 214), (Scrum, Iter. Dev.: 260) }
319
DevOps
{ (Iter. Dev.: 280), (Kanban: 251), (Kanban, Scrum: 236), (Scrum: 294), (Iter. Dev., Scrum: 260), (Kanban, Iter. Dev.: 224), (Scrum, Iter. Dev., Kanban: 212) }
303
eXtreme Programming
{ (Iter. Dev.: 270), (Scrum: 277), (Kanban: 232), (Scrum, Kanban: 220) (Scrum, Iter. Dev.: 247), (Kanban, Iter. Dev.: 209) }
243
Lean Software Development
{ (Iter. Dev.: 225), (Scrum: 220), (Scrum, Iter. Dev.: 206) }
218
Feature Driven Development
{ (Iter. Dev.: 208) }

**FIGURE 4** Base methods and method combinations (35% threshold) in the non-filtered dataset (left) and on the dataset filtered for intentional combination of different development frameworks and methods (right). The figures reads from the right to the left, e.g., 309 participants use Scrum, Kanban and DevOps in combination (all data points). The lower part of the figure (dashed arrows) shows the other/alternative entry points for those combinations that are part of the combinations shown in the upper part of the figure.

in Figure 4 *do not* provide the full picture as they only form the "core", but are complemented with further frameworks, methods, and practices, which will be elaborated in more detail in the following sections.

> **Finding 2:** Among the 24 frameworks and methods presented to the study participants, we identified 17 (entire dataset, Figure 4, left) and 27 (dataset filtered for question PU04, Figure 4, right) core groups with two to four elements for which the study participants agree with at least 35% on their combined use. These combinations are based on eight *base methods* that provide the frame for hybrid development methods.

## 4.3 | Combined Use of Frameworks, Methods, and Practices

The second step in the quantitative analysis is the investigation of the combined use of frameworks, methods *and* practices (Figure 3, PU09 and PU10). Of the 1,467 data points, 769 provide data. As shown in Figure 3, questions PU09 and PU10 provided 36 items each to choose from and a free-text option. For each of the 36 items, we used a Likert-scale to rate the use *and* the frequency of use. The following analyses are based on those answers that we categorized into the category "Use" (Figure 3). In total, we used 769 data points for analysis of which 742 had multiple selections.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

As described in Section 3.3.3, to analyze the combinations of practices within the base methods and method combinations provided in Section 4.2, we used an 85% threshold for the agreement regarding the combined use. That is, for each method combination identified in Section 4.2, the combinations of practices within these have been computed. All analyses were performed using base methods and method combinations resulting from the entire dataset and from the filtered dataset based on the answers to the question PU04 (Figure 3). The overall result is shown in Figure 6, which will be described step by step in the following subsections.

### 4.3.1 | Unfiltered Practices

As a first step, the (non-filtered) dataset was analyzed for the most commonly used practices, i.e., those practices with the highest agreement regarding combined use without a particular combination of methods. To find these practice combinations and to find those groups that have the largest agreement in the entire dataset, we explored the dataset. The smallest group with the highest agreement in the data was the pair *Code Review* and *Coding Standards* (n=674, agreement=0.87). The agreement level of 0.87 was also used to set the threshold of 85% agreement as introduced in Section 3.3.3.

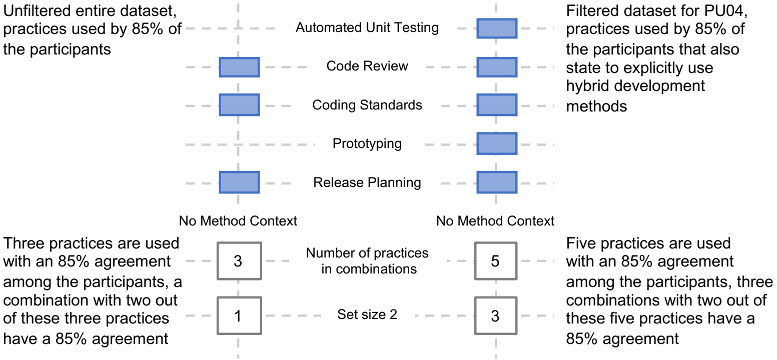

Unfiltered entire dataset, practices used by 85% of the participants
Automated Unit Testing
Code Review
Coding Standards
Prototyping
Release Planning
Filtered dataset for PU04, practices used by 85% of the participants that also state to explicitly use hybrid development methods
No Method Context
No Method Context
Three practices are used with an 85% agreement among the participants, a combination with two out of these three practices have a 85% agreement
3
1
Number of practices in combinations
Set size 2
5
3
Five practices are used with an 85% agreement among the participants, three combinations with two out of these five practices have a 85% agreement

**FIGURE 5** Overview of the most frequently used practices in the entire dataset (left) and in the filtered dataset (PU04, right). The figure illustrates the most frequently used practices and also shows how many possible combinations can be found with 85% agreement among the participants.

The results of the analyses of the entire (non-filtered) dataset using the 85% threshold are shown in Figure 5. The figure shows for the entire dataset three practices (*Code Review*, *Coding Standards* and *Release Planning* with an 85% agreement) for which one pair of two practices with an 85% agreement could be find. Likewise, in the filtered dataset (after applying PU04 as filter), five practices could be identified (*Code Review*, *Coding Standards*, *Release Planning*, *Automated Unit Testing* and *Protoyping* with an 85% agreement). Of these five practices, three pairs of two composed from the five practices could be identified, which have at least 85% agreement among the participants of the study.

### 4.3.2 | Individual Practices

In the same reading as for Figure 5, the upper part of Figure 6 presents the practices reaching 85% agreement within the context of the respective base methods and method combinations. The upper-left part of Figure 6 presents the results for the entire (non-filtered) dataset, while the upper-right part of Figure 6 presents the results for those base methods and method combinations computed from the filtered dataset after applying PU04 as a filter (Section 4.2).

For each practice (Figure 3; PU10, 36 items to choose from), Figure 6 shows the assignment to a base method or a method combination for which 85% agreement could be found in the dataset. The total number of such practices assigned to a particular method combination is shown in the row "Number of practices in combinations" beneath the respective method combinations. For instance, for the method combination *(Scrum–Kanban)*, 14 practices are assigned to this method combination in the entire dataset and, respectively, 15 practices are assigned to this method combination in the PU04-filtered dataset. All possible combinations of frameworks, methods, and practices with 85% agreement are constructed from these individual practices, which is elaborated in more detail in Section 4.3.3. The visualization in the upper part of Figure 6 allows for two main observations:

**1$^{st}$ Observation:** The sparsity of rows and thus a limited number of practices consistently selected by the participants in the context of a given method or method combination.

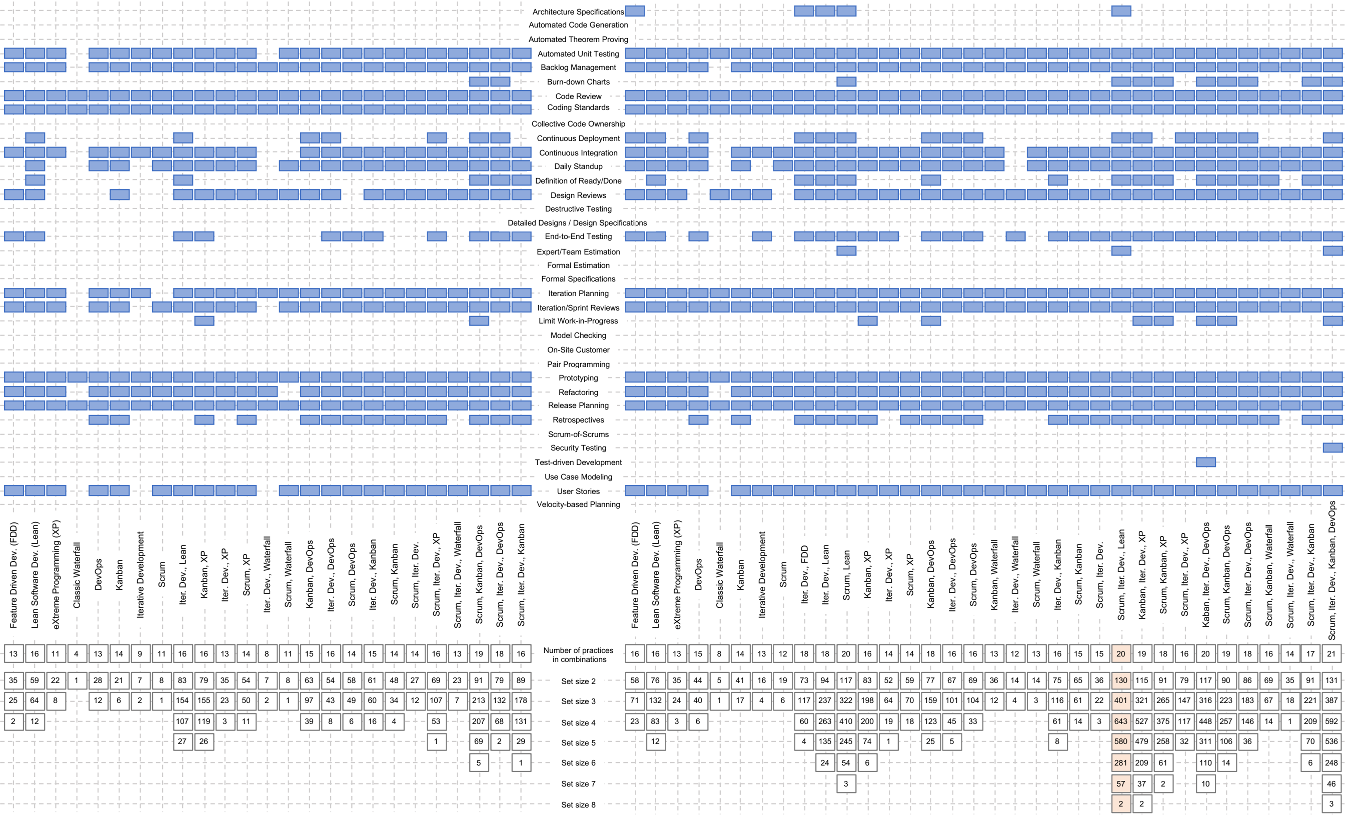
©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

**2nd Observation:** The selected practices (highlighted rows) are consistent across different method combinations. That is, a limited number of practices is consistently used with an agreement of at least 85% regardless of the actual method combination.

In addition, two minor observations can be made: first, the "density" of the practices for which the participants agree regarding their combined use is higher in the PU04-filtered dataset than in the non-filtered dataset, i.e., in that share of the data in which the participants explicitly stated to combine multiple frameworks, methods, and practices. Second, it seems that as if the larger the number of combined methods is the more practices find an agreement among the participants. For instance, the rightmost method combination in Figure 6 *(Scrum–Iterative Development–Kanban–DevOps)* has 21 practices assigned for which the participants find an agreement of at least 85%.

### 4.3.3 | Combinations of Practices

The lower part of Figure 6 extends the analysis from Section 4.3.2 starting with the "Number of practices in combinations" row. This row shows how many practices are assigned to the different method combinations thus forming the basis for framework, method, and practice combinations to derive hybrid development methods. Within these sets of practices, we search for *practice tuples* of increasing size having an agreement of at least 85% and that are used in combination in the respective method combination.



| | Feature Driven Dev. (FDD) | Lean Software Dev. (Lean) | eXtreme Programming (XP) | Classic Waterfall | DevOps | Kanban | Iterative Development | Scrum | Iter. Dev., Lean | Kanban, XP | Iter. Dev., XP | Scrum, XP | Iter. Dev., Waterfall | Scrum, Waterfall | Kanban, DevOps | Iter. Dev., DevOps | Scrum, DevOps | Iter. Dev., Kanban | Scrum, Kanban | Scrum, Iter. Dev. | Scrum, Iter. Dev., XP | Scrum, Iter. Dev., Waterfall | Scrum, Kanban, DevOps | Scrum, Iter. Dev., DevOps | Scrum, Iter. Dev., Kanban |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Number of practices in combinations | 13 | 16 | 11 | 4 | 13 | 14 | 9 | 11 | 16 | 16 | 13 | 14 | 8 | 11 | 15 | 16 | 14 | 15 | 14 | 14 | 16 | 13 | 19 | 18 | 16 |
| Set size 2 | 35 | 59 | 22 | 1 | 28 | 21 | 7 | 8 | 83 | 79 | 35 | 54 | 7 | 8 | 63 | 54 | 58 | 61 | 48 | 27 | 69 | 23 | 91 | 79 | 89 |
| Set size 3 | 25 | 64 | 8 | | 12 | 6 | 2 | 1 | 154 | 155 | 23 | 50 | 2 | 1 | 97 | 43 | 49 | 60 | 34 | 12 | 107 | 7 | 213 | 132 | 178 |
| Set size 4 | 2 | 12 | | | | | | | 107 | 119 | 3 | 11 | | | 39 | 8 | 6 | 16 | 4 | | 53 | | 207 | 68 | 131 |
| Set size 5 | | | | | | | | | 27 | 26 | | | | | | | | | | | 1 | | 69 | 2 | 29 |
| Set size 6 | | | | | | | | | | | | | | | | | | | | | | | 5 | | 1 |
| Set size 7 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Set size 8 | | | | | | | | | | | | | | | | | | | | | | | | | |

| | Feature Driven Dev. (FDD) | Lean Software Dev. (Lean) | eXtreme Programming (XP) | DevOps | Classic Waterfall | Kanban | Iterative Development | Scrum | Iter. Dev., FDD | Iter. Dev., Lean | Scrum, Lean | Kanban, XP | Iter. Dev., XP | Scrum, XP | Kanban, DevOps | Iter. Dev., DevOps | Scrum, DevOps | Kanban, Waterfall | Iter. Dev., Waterfall | Scrum, Waterfall | Iter. Dev., Kanban | Scrum, Kanban | Scrum, Iter. Dev. | Scrum, Iter. Dev., Lean | Kanban, Iter. Dev., XP | Scrum, Kanban, XP | Scrum, Iter. Dev., XP | Kaban, Iter. Dev., DevOps | Scrum, Kanban, DevOps | Scrum, Iter. Dev., DevOps | Scrum, Kanban, Waterfall | Scrum, Iter. Dev., Waterfall | Scrum, Iter. Dev., Kanban | Scrum, Iter. Dev., Kanban, DevOps |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Number of practices in combinations | 16 | 16 | 13 | 15 | 8 | 14 | 13 | 12 | 18 | 18 | 20 | 16 | 14 | 14 | 18 | 16 | 16 | 13 | 12 | 13 | 16 | 15 | 15 | 20 | 19 | 18 | 16 | 20 | 19 | 18 | 16 | 14 | 17 | 21 |
| Set size 2 | 58 | 76 | 35 | 44 | 5 | 41 | 16 | 19 | 73 | 94 | 117 | 83 | 52 | 59 | 77 | 67 | 69 | 36 | 14 | 14 | 75 | 65 | 36 | 130 | 115 | 91 | 79 | 117 | 90 | 86 | 69 | 35 | 91 | 131 |
| Set size 3 | 71 | 132 | 24 | 40 | 1 | 17 | 4 | 6 | 117 | 237 | 322 | 198 | 64 | 70 | 159 | 101 | 104 | 12 | 4 | 3 | 116 | 61 | 22 | 401 | 321 | 265 | 147 | 316 | 223 | 183 | 67 | 18 | 221 | 387 |
| Set size 4 | 23 | 83 | 3 | 6 | | | | | 60 | 263 | 410 | 200 | 19 | 18 | 123 | 45 | 33 | | | | 61 | 14 | 3 | 643 | 527 | 375 | 117 | 448 | 257 | 146 | 14 | 1 | 209 | 592 |
| Set size 5 | | 12 | | | | | | | 4 | 135 | 245 | 74 | 1 | | 25 | 5 | | | | | 8 | | | 580 | 479 | 258 | 32 | 311 | 106 | 36 | | | 70 | 536 |
| Set size 6 | | | | | | | | | | 24 | 54 | 6 | | | | | | | | | | | | 281 | 209 | 61 | | 110 | 14 | | | | 6 | 248 |
| Set size 7 | | | | | | | | | | | 3 | | | | | | | | | | | | | 57 | 37 | 2 | | 10 | | | | | | 46 |
| Set size 8 | | | | | | | | | | | | | | | | | | | | | | | | 2 | 2 | | | | | | | | | 3 |

**FIGURE 6** Overview of the practices used with 85% agreement in hybrid development methods. The left part of the figure shows the analysis results for the entire, non-filtered dataset, the right part illustrates the analysis results for the filtered dataset based on the value of question PU04. The upper part of the figure shows the practices used together with the different method combinations, and the lower part shows the of possible combinations of practices of given set sizes within the different method combinations, based on the *number of practices in combinations*-row.

Taking the combination *(Scrum–Kanban)* as an example, 14 (entire dataset) and 15 (PU04-filtered dataset) practices are assigned to this combination. In the first step, we search for pairs of two practices from these 14 (15) practices with the required agreement level. This results in 48 pairs (for the entire dataset) and, respectively, 65 pairs (PU04-filtered dataset) of practices out of 14 (15) practices. In the next step, we search for 3-tuples, then for 4-tuples, and so forth until no *x*-tuple with the required agreement level is found. As Figure 6 shows, the biggest set size with

ⓒWiley. PREPRINT. This is the author's version of the work. Not for redistribution.

an agreement of at least 85% is eight. For instance, in the PU04-filtered dataset and for the combination *(Scrum–Iterative Development–Kanban–DevOps)*, three 8-tuples of practices from the 21 practices assigned to this combination can be found in the dataset. Hence, the lower part of Figure 6 allows for two main observations:

**1st Observation:** The larger the combination size, i.e., the more practices are included in the set of practices individually reaching the required 85% agreement, the more "general" agreement can be found regarding the practices that would be included in a method or method combination.

**2nd Observation:** The larger the number of combinations within a group the more practices are consistently selected by the participants. This provides a quantitative description of the process variants within a specific hybrid method.

Also, similar to the observations from Section 4.3.2, we see that the larger the number of combined methods, the more combinations of practices find agreement among the practitioners, and the size and the amount of combinations are bigger in the PU04-filtered data than in the entire dataset.

**Finding 3:** In Summary, analyzing the 36 practices and their relation to the methods and method combinations found in Section 4.2, we find few practices only that find agreement of at least 85% among practitioners. However, as shown in Figure 6 the assignments show a consistency across the base methods and method combinations. That is, few practices only are consistently used for hybrid methods, yet, allow for framing a large number of process variants.

## 4.4 | Investigating Process Variants

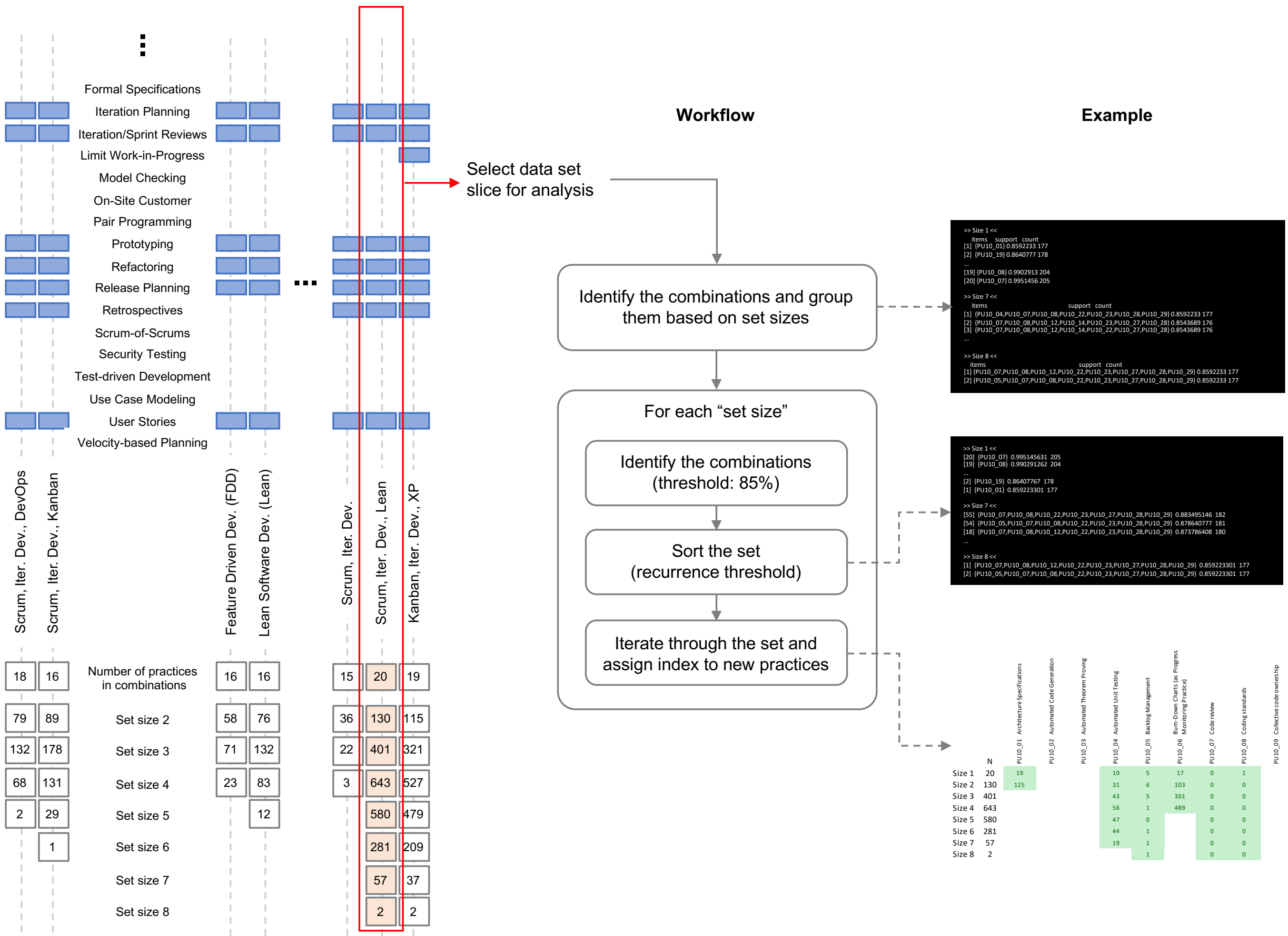
The clusters of frameworks, methods, and practices as presented in Figure 5 and Figure 6 (Section 4.3.3) only show *which* frameworks, methods, and practices are potentially combined with one another. However, all practices involved in theses analyses reached an agreement level of at least 85% and, thus, it is necessary to provide a better characterization of when a particular practice is included in a combination, i.e., to rank the individual practices.

For this, we "re-apply" our data analysis procedure to determine the practices' ranks. Figure 7 outlines our approach, which is discussed in detail in this section. To showcase our approach, we selected the method combination that yielded the highest number of potentially relevant combinations, i.e., the method combination *(Scrum–Iterative Development–Lean Software Development)* for which up to 643 process variants can be constructed. Having selected this method combination, we apply the following procedure (as illustrated in Figure 7):

1. Filter the dataset to isolate those data points related to the hybrid method under investigation (i.e., *(Scrum–Iterative Development–Lean)* in our case) and for which participants answered PU04 with "Yes".
2. Identify the combinations, specifically, the combinations with their growing set sizes (Figure 6 lower part).
3. For each present combination and set size (i.e., 1 to 8 in our case) do:
   (a) Identify the combinations with a recurrence threshold of at least 85%.
   (b) Sort the set based on recurrence threshold.
   (c) Starting from the combination with the highest recurrence threshold, iterate on the sorted set to identify any new practice and associate to the practice the index of the combination.

Figure 8 visualizes the results of the analysis. In the figure, rows are internally ordered based on the sorting established for set size 4, which is the combinations size with the highest number of variants (i.e., 643). For each set size, the first row reports the index of the first variant in which a given practice is detected, while the second row indicates the agreement on the process variant within the data set. That is, for the selected case, the practices *Code Review*, *Coding Standards*, *Refactoring*, and *Release Planning* appear "immediately" just in the starting configuration. Just in the next process variant, *Backlog Management* is added, in the next one *Prototyping* is added, and so forth. After *Iteration Planning*, in this combination, there are six more combinations of the up to seven practices found so far, before *Iteration/Sprint Planning* is added as new practice in the 11th combination. Likewise, 260 combinations including up to 15 practices appear in the ranked list of practices before the *Definition of Done/Ready* appears for the first time. From this, we can conclude the relevance of a specific practice in a hybrid method. For instance, in sets of size 4, we can see that *Burn-down Charts*, *Continuous Deployment*, and *Expert/Team-based Estimation* are practices mentioned only in combinations at the end of

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.



| | Scrum, Iter. Dev., DevOps | Scrum, Iter. Dev., Kanban | Feature Driven Dev. (FDD) | Lean Software Dev. (Lean) | Scrum, Iter. Dev. | Scrum, Iter. Dev., Lean | Kanban, Iter. Dev., XP |
|---|---|---|---|---|---|---|---|
| Number of practices in combinations | 18 | 16 | 16 | 16 | 15 | 20 | 19 |
| Set size 2 | 79 | 89 | 58 | 76 | 36 | 130 | 115 |
| Set size 3 | 132 | 178 | 71 | 132 | 22 | 401 | 321 |
| Set size 4 | 68 | 131 | 23 | 83 | 3 | 643 | 527 |
| Set size 5 | 2 | 29 | | 12 | | 580 | 479 |
| Set size 6 | | 1 | | | | 281 | 209 |
| Set size 7 | | | | | | 57 | 37 |
| Set size 8 | | | | | | 2 | 2 |

```
>> Size 1 <<
   items     support   count
[1]  {PU10_01} 0.8592233 177
[2]  {PU10_19} 0.8640777 178
...
[19] {PU10_08} 0.9902913 204
[20] {PU10_07} 0.9951456 205

>> Size 7 <<
   items                                                    support   count
[1]  {PU10_04,PU10_07,PU10_08,PU10_22,PU10_23,PU10_28,PU10_29} 0.8592233 177
[2]  {PU10_07,PU10_08,PU10_12,PU10_14,PU10_23,PU10_27,PU10_28} 0.8543689 176
[3]  {PU10_07,PU10_08,PU10_12,PU10_14,PU10_22,PU10_27,PU10_28} 0.8543689 176
...

>> Size 8 <<
  items                                                         support   count
[1] {PU10_07,PU10_08,PU10_12,PU10_22,PU10_23,PU10_27,PU10_28,PU10_29} 0.8592233 177
[2] {PU10_05,PU10_07,PU10_08,PU10_22,PU10_23,PU10_27,PU10_28,PU10_29} 0.8592233 177
```

```
>> Size 1 <<
[20]  {PU10_07}  0.995145631  205
[19]  {PU10_08}  0.990291262  204
...
[2]  {PU10_19}  0.86407767  178
[1]  {PU10_01}  0.859223301  177

>> Size 7 <<
[55]  {PU10_07,PU10_08,PU10_22,PU10_23,PU10_27,PU10_28,PU10_29}  0.883495146  182
[54]  {PU10_05,PU10_07,PU10_08,PU10_22,PU10_23,PU10_28,PU10_29}  0.878640777  181
[18]  {PU10_07,PU10_08,PU10_12,PU10_22,PU10_23,PU10_28,PU10_29}  0.873786408  180
...

>> Size 8 <<
[1]  {PU10_07,PU10_08,PU10_12,PU10_22,PU10_23,PU10_27,PU10_28,PU10_29}  0.859223301  177
[2]  {PU10_05,PU10_07,PU10_08,PU10_22,PU10_23,PU10_27,PU10_28,PU10_29}  0.859223301  177
```

| | N | PU10_01 Architecture Specifications | PU10_02 Automated Code Generation | PU10_03 Automated Theorem Proving | PU10_04 Automated Unit Testing | PU10_05 Backlog Management | PU10_06 Burn-Down Charts (as Progress Monitoring Practice) | PU10_07 Code review | PU10_08 Coding standards | PU10_09 Collective code ownership |
|---|---|---|---|---|---|---|---|---|---|---|
| Size 1 | 20 | 19 | | | 10 | 5 | 17 | 0 | 1 | |
| Size 2 | 130 | 125 | | | 31 | 6 | 103 | 0 | 0 | |
| Size 3 | 401 | | | | 43 | 5 | 301 | 0 | 0 | |
| Size 4 | 643 | | | | 56 | 1 | 489 | 0 | 0 | |
| Size 5 | 580 | | | | 47 | 0 | | 0 | 0 | |
| Size 6 | 281 | | | | 44 | 1 | | 0 | 0 | |
| Size 7 | 57 | | | | 19 | 1 | | 0 | 0 | |
| Size 8 | 2 | | | | | 1 | | 0 | 0 | |

**FIGURE 7** Illustration of the basic procedure to analyze and rank the different process variants. The figure also shows the selected hybrid method from Figure 6 for which we showcase the analysis procedure.

the sorted set (i.e., indexes of 489, 490, and 563 on a set with 643 process variants), which makes them less relevant for constructing a hybrid method in this context.

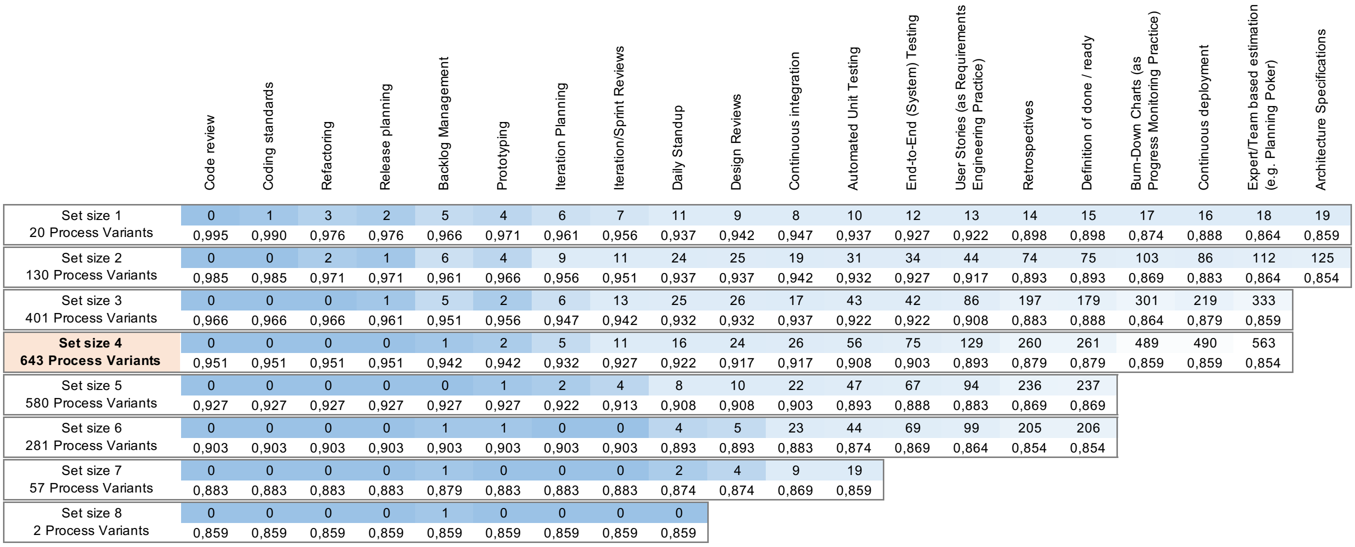
Looking at the highlighted row of Figure 8, it can be observed how some practices are much more appealing to the participants as occurring within the variants that have the highest recurrence threshold, which leads to the following two observations:

**1st Observation:** Practices on the left side are elements that characterize the hybrid process variant under investigation, while practices on the right side are less relevant in the hybrid method.

**2nd Observation:** Even though all rows in Figure 8 follow the ordering derived from the variants in the set of size 4, the sorting on the other set sizes is rather accurate and aligned.

Hence, Figure 8 does not only show which frameworks, methods, and practices are combined, but also the "preferred" practices used to embody the base methods and/or the method combinations framing hybrid methods as identified in Section 4.2. Even with the high level of agreement of 85% regarding the combined use, Figure 8 shows that there are practices with a higher agreement among the practitioners, and these agreement levels together with the rank of a practice (the index of its first appearance in the sorted list) indicate the more "prominent" hybrid methods. On the other hand, the figure also shows that the increasing size of the practice cluster does not necessarily lead to a high variability (see set size 8, which only has two variants that are characterized by the presence of *Backlog Management* or the lack thereof).

ⒸWiley. PREPRINT. This is the author's version of the work. Not for redistribution.



| | Code review | Coding standards | Refactoring | Release planning | Backlog Management | Prototyping | Iteration Planning | Iteration/Sprint Reviews | Daily Standup | Design Reviews | Continuous integration | Automated Unit Testing | End-to-End (System) Testing | User Stories (as Requirements Engineering Practice) | Retrospectives | Definition of done / ready | Burn-Down Charts (as Progress Monitoring Practice) | Continuous deployment | Expert/Team based estimation (e.g. Planning Poker) | Architecture Specifications |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Set size 1 | 0 | 1 | 3 | 2 | 5 | 4 | 6 | 7 | 11 | 9 | 8 | 10 | 12 | 13 | 14 | 15 | 17 | 16 | 18 | 19 |
| 20 Process Variants | 0,995 | 0,990 | 0,976 | 0,976 | 0,966 | 0,971 | 0,961 | 0,956 | 0,937 | 0,942 | 0,947 | 0,937 | 0,927 | 0,922 | 0,898 | 0,898 | 0,874 | 0,888 | 0,864 | 0,859 |
| Set size 2 | 0 | 0 | 2 | 1 | 6 | 4 | 9 | 11 | 24 | 25 | 19 | 31 | 34 | 44 | 74 | 75 | 103 | 86 | 112 | 125 |
| 130 Process Variants | 0,985 | 0,985 | 0,971 | 0,971 | 0,961 | 0,966 | 0,956 | 0,951 | 0,937 | 0,937 | 0,942 | 0,932 | 0,927 | 0,917 | 0,893 | 0,893 | 0,869 | 0,883 | 0,864 | 0,854 |
| Set size 3 | 0 | 0 | 0 | 1 | 5 | 2 | 6 | 13 | 25 | 26 | 17 | 43 | 42 | 86 | 197 | 179 | 301 | 219 | 333 | |
| 401 Process Variants | 0,966 | 0,966 | 0,966 | 0,961 | 0,951 | 0,956 | 0,947 | 0,942 | 0,932 | 0,932 | 0,937 | 0,922 | 0,922 | 0,908 | 0,883 | 0,888 | 0,864 | 0,879 | 0,859 | |
| **Set size 4** | 0 | 0 | 0 | 0 | 1 | 2 | 5 | 11 | 16 | 24 | 26 | 56 | 75 | 129 | 260 | 261 | 489 | 490 | 563 | |
| **643 Process Variants** | 0,951 | 0,951 | 0,951 | 0,951 | 0,942 | 0,942 | 0,932 | 0,927 | 0,922 | 0,917 | 0,917 | 0,908 | 0,903 | 0,893 | 0,879 | 0,879 | 0,859 | 0,859 | 0,854 | |
| Set size 5 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 4 | 8 | 10 | 22 | 47 | 67 | 94 | 236 | 237 | | | | |
| 580 Process Variants | 0,927 | 0,927 | 0,927 | 0,927 | 0,927 | 0,927 | 0,922 | 0,913 | 0,908 | 0,908 | 0,903 | 0,893 | 0,888 | 0,883 | 0,869 | 0,869 | | | | |
| Set size 6 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 4 | 5 | 23 | 44 | 69 | 99 | 205 | 206 | | | | |
| 281 Process Variants | 0,903 | 0,903 | 0,903 | 0,903 | 0,903 | 0,903 | 0,903 | 0,903 | 0,893 | 0,893 | 0,883 | 0,874 | 0,869 | 0,864 | 0,854 | 0,854 | | | | |
| Set size 7 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 4 | 9 | 19 | | | | | | | | |
| 57 Process Variants | 0,883 | 0,883 | 0,883 | 0,883 | 0,879 | 0,883 | 0,883 | 0,883 | 0,874 | 0,874 | 0,869 | 0,859 | | | | | | | | |
| Set size 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | | | | | | | | | | | |
| 2 Process Variants | 0,859 | 0,859 | 0,859 | 0,859 | 0,859 | 0,859 | 0,859 | 0,859 | 0,859 | | | | | | | | | | | |

**FIGURE 8** Characterization of the variants of the *(Scrum–Iterative Development–Lean Software Development)* hybrid method. Rows are internally ordered based on the sorting established for set size 4, i.e., the combinations size with the highest number of variants and the running example used in the text.

**Finding 4:** Analyzing the variance of a specific hybrid method through the procedure outlined in Figure 7, it was possible to identify and sort a small set of practices that characterize the given hybrid method based on the participants' selections. Moreover, no matter the threshold chosen with regards to the agreement level among participants' responses, a noticeable consistency can be seen in the ordering of the practices, which indicates the presence of a core set of practices.

## 4.5 | Constructing Hybrid Development Methods

The analyses conducted in the Sections 4.2 to 4.4 provide important insights regarding the *base methods*, the basic *method combinations* and the *number of practices assigned* to these base methods and method combinations. In this section, we demonstrate how to utilize our analysis method to incrementally construct hybrid development methods. For this, we apply the following procedure:

1. Based on the smallest groups of frameworks and methods for the entire dataset and for the PU04-filtered dataset shown in Figure 4, we form the "umbrella" to construct our hybrid method. Our current data allows for constructing 17 groups (all data) and, respectively, 27 groups (PU04-filtered data), which are constructed from eight frameworks and methods in total.

2. Based on the smallest groups of practices for the entire dataset and for the PU04-filtered dataset, which are shown in Figure 5, we form the "core" of practices from the different pairs. Our current data allows for constructing one pair for the entire dataset and three for the PU04-filtered dataset.

3. For each base method or method combination created in the first step, we add the core(s) as created in the section step to set an extended method context. For instance, instead of looking for all practices to be combined with *Scrum*, we search all additional practices meeting the required agreement level of 85% for the new combination *(Scrum–core$_i$)* with i denoting the cores identified.

4. In the final step, we integrate all frameworks, methods, and practices into hybrid methods by building the unique combinations (the process variants) of all these components. Applying our analysis from Section 4.4, we can read the "promising" variants for our hybrid method. Hence, we can create all process variants and rank them using the participants' level of agreement.

That is, we aim to identify those practices (if any) that are included in bigger combinations containing the "core" for each of the base methods or method combinations. We applied this procedure to both the entire dataset and the PU04-filtered dataset, which results in the *(method-core$_i$-practice)* combinations shown in Figure 9 (left) for the entire dataset and Figure 9 (right) for the PU04-filtered dataset.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

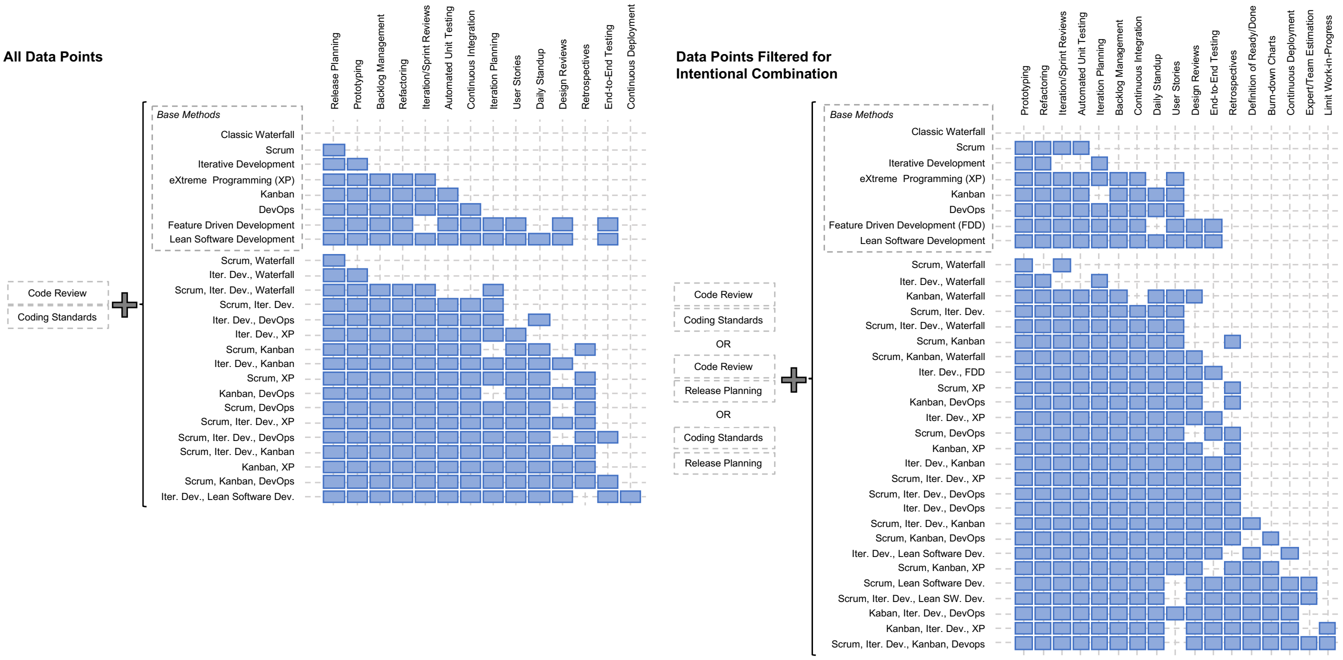


**FIGURE 9** Combinations resulting from applying the statistical analysis method: The right part of the figure characterizes the base methods (top) and method combinations (bottom) generated by identifying recurring combinations of three practices with at least 85% agreement level that contain the core of the entire non-filtered dataset, i.e., (*Code Review–Coding Standards*). The right part of the figure shows a refined characterization using the dataset filtered for question PU04 of base methods (top) and method combinations (bottom). The three cores used are: (*Code Review–Coding Standards*), (*Coding Standards–Release Planning*), (*Code Review–Release Planning*).

## Constructing the "Water-Scrum-Fall"

Taking the *Classic Waterfall* as an example, we see in both parts of Figure 9 that the *Waterfall* is characterized by the core-practices only. Moving on to "Water-Scrum-Fall", i.e., the combination *(Scrum–Waterfall)*, in the entire dataset (Figure 9, left), we see one single combination of size three containing the *(Scrum–Waterfall)* method combination, the core consisting of *(Code Review–Coding Standards)* and *Release Planning* as third practice. In the PU04-filtered data (Figure 9, right), "Water-Scrum-Fall" is characterized by the *(Scrum–Waterfall)* method combination and three cores of which either is extended by (*Prototyping–Iteration/Sprint Reviews*) as illustrated below:

$$(\textit{Scrum–Waterfall}) + \overbrace{\begin{bmatrix} (\textit{Code Review–Coding Standards}) \\ (\textit{Code Review–Release Planning}) \\ (\textit{Coding Standards–Release Planning}) \end{bmatrix}}^{\textbf{XOR}} + (\textit{Prototyping–Iteration/Sprint Reviews})$$

Both parts of Figure 9 also show the gradual increase of the practice pools. Staying with *Scrum* in the entire dataset, i.e., (*Scrum–Code Review–Coding Standards–Release Planning*), the base method *Iterative Development* "extends" this combination with *Prototyping*, i.e., the combination would be (*Iterative Development–Code Review–Coding Standards–Release Planning–Protyping*). That is, *Scrum* is characterized by the practice combination (*Code Review–Coding Standards–Release Planning*) and *Iterative Development* is characterized by (*Code Review–Coding Standards–Release Planning–Prototyping*).

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

### Constructing a Hybrid Method from the Variant Space

As Figure 9 only shows which frameworks, methods, and practices to combine, we use our findings from Section 4.4 to provide a recommendation, based on the practices' rank. Below, the first quadruple from Figure 8 is outlined:

$$(\mathit{Scrum}\text{–}\mathit{Iterative\ Development}\text{–}\mathit{Lean\ SWDev.}) + \overbrace{\left[\begin{array}{l} \mathit{Code\ Review} \\ \mathit{Coding\ Standards} \\ \mathit{Refactoring} \\ \mathit{Release\ Planning} \end{array}\right]_{a_4=0.951}}^{\text{All}}$$

It can be seen that the overall agreement level for this practice cluster is $a_4 = 0.951$, where $a_4$ points to the "initial" cluster of practices with a set size of 4 from Figure 8. Moving on to the first index in Figure 8, *Backlog Management* is added. To keep the quadruples, i.e., implementing the requirement of including four practices only in our hybrid method, instead of just reusing the set above, in this case, we have to limit the selection accordingly to three, allowing *Backlog Management* to be added as forth:

$$(\mathit{Scrum}\text{–}\mathit{Iterative\ Development}\text{–}\mathit{Lean\ SWDev.}) + \overbrace{\left[\begin{array}{l} \mathit{Code\ Review}_{a_3=0.966} \\ \mathit{Coding\ Standards}_{a_3=0.966} \\ \mathit{Refactoring}_{a_3=0.966} \\ \mathit{Release\ Planning}_{a_3=0.961} \end{array}\right]_{a_{4\to 3}=[0.951,0.966]}}^{\text{3 out of 4}} + (\mathit{Backlog\ Management}_{a_4=0.942})$$

This has several consequences: first, instead of quadruples, we now have to build *triplets* from the four available practices. That is, for the practices, we have to move from the set size 4 to the set size 3 (Figure 8), which impacts the level of agreement. To make this visible, the agreement level is added to the practices, and $a_3$ points to the set size 3. Second, the boundaries for the agreement level changes, which is shown by the index $a_4 \to a_3 = [0.951, 0.966]$, where the lower bound is taken from the lowest $a_4$−agreement level and the upper bound is taken from the highest $a_3$-agreement level.

Yet, this is the first selection step only. The next consequence is the necessity for re-computing the specific agreement levels for the resulting triplets. Therefore, the algorithm described in Section 4.4 needs to be re-executed for the triplets. In total, we can construct four triplets from the four given practices. The result of the algorithm's execution is shown in Table 4. The outcome, i.e., the minimal level of agreement is now added to the formal representation ($a_3 \geq 0.956$), which means that every triplet has at least an agreement level of 95.6%.

**TABLE 4** Results for re-computing the agreement levels for triplets in the construction of a hybrid method. The computed minimal level of agreement for three out of four practices is $a_3 = 0.95631068$.

| Combination | Code Review<br>PU10_07 | Coding Standards<br>PU10_08 | Refactoring<br>PU10_28 | Release Planning<br>PU10_29 | Agreement<br>for $a_3$ |
|---|---|---|---|---|---|
| Combination 1 | X | X | X | | 0.966019417 |
| Combination 2 | X | X | | X | 0.961165049 |
| Combination 3 | | X | X | X | 0.95631068 |
| Combination 4 | X | | X | X | 0.95631068 |

The next step is now adding the practice *Backlog Management*. We are going back to four practices, i.e., we have to re-compute the agreement levels with quadruples again, but, instead of using the "initial" four practices, we now used the triplets and the practice *Backlog Management*. Re-computing the new quadruples results in a minimal level of agreement $a_4 = 0.932038835$ as shown in Table 5. That is, selecting one triplet from the "initial" four practices and adding *Backlog Management* as a fourth practice results into an overall minimal agreement level of 93.2%.

Having executed the different computations, the final hybrid method can be constructed as shown below: The starting point is the method combination (*Scrum*–*Iterative Development*–*Lean Software Development*), which is complemented by three out of the four initial practices that provide a minimal agreement level of $a_3 \geq 0.956$ and, finally, the practice *Backlog Management* is added, which results into a final minimal agreement level

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

**TABLE 5** Results for re-computing the agreement levels for quadruples in the construction of a hybrid method. The computed minimal level of agreement for three out of four practices and *Backlog Management* is $a_4 = 0.932038835$.

| **Combination** | **Code Review** | **Coding Standards** | **Refactoring** | **Release Planning** | **Backlog Management** | **Agreement** |
|---|---|---|---|---|---|---|
| | **PU10_07** | **PU10_08** | **PU10_28** | **PU10_29** | **PU10_05** | **for $a_4$** |
| Combination 1 | X | X | X | | X | 0.941747573 |
| Combination 2 | X | X | | X | X | 0.941747573 |
| Combination 3 | | X | X | X | X | 0.932038835 |
| Combination 4 | X | | X | X | X | 0.932038835 |

of $a_4 \geq 0.932$.

$$(\textit{Scrum–Iterative Development–Lean SWDev.}) + \overbrace{\begin{bmatrix} \textit{Code Review} \\ \textit{Coding Standards} \\ \textit{Refactoring} \\ \textit{Release Planning} \end{bmatrix}_{a_3 \geq 0.956}}^{\textbf{3 out of 4}} + (\textit{Backlog Management}_{a_4 \geq 0.932})$$

**Finding 5:** Applying our analysis procedures, we can define a statistical construction procedure for describing hybrid methods. Also, our description is not only limited to specific hybrid method instances; we can also characterize a hybrid method *and* its process variants.

## 5 | DISCUSSION

Having presented our results, we conclude this paper by answering the research questions, discussing our findings and discussing the threats to validity.

### 5.1 | Answering the Research Questions

The findings provide a rich quantitative basis and evidence to answer our research questions posed in Section 3.1:

**RQ1:** Klünder et al.[16] found that processes mainly *evolve* into hybrid methods, and provided evidence and generalized the claim by West et al.[9] about the *"Water-Scrum-Fall"*. With this paper, we provide insights regarding eight base methods that are recurrently combined to form hybrid development methods (cf. Finding 2).

**RQ2:** In this paper, we identify the most frequently used practices and how these are combined with each other. Our results reveal a small *core* of practices used by practitioners regardless of the (hybrid) development method selected (cf. Finding 3).

**RQ3:** Analyzing the variability within the hybrid development methods identified, it is possible to identify small sets of practices that characterize each hybrid method. These practices are consistently mentioned in relation to the hybrid, indicating process variants that are clearly preferred over others (cf. Finding 4).

**RQ4:** In modern software and system development, methods and practices are often combined into processes that are context-dependent. However, when attempting to characterize the different methods by systematically constructing the set of practices using a bottom-up strategy, we show that the resulting combinations of practices vary very little and consistently repeat the same practices (cf. Finding 5).

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

## 5.2 | Hyped Methods and Old Practices

So far, we could identify eight base methods and few practices that, together, are the heart of hybrid development methods. Figure 9 shows these key components that find agreement across the participants of the HELENA study. Looking closer at the practices, we see that hybrid methods are heavily composed of (mostly technical) practices that have been used in software development for decades, namely *Code Review*, *Coding Standards* and *Release Planning*. On the one hand, the sets of practices and their assignment to methods show obvious similarities as shown in Figure 6. On the other hand, it is not trivial to identify a specific method or method combination from the practices used in a specific context.

Accepting that formally defined methods are not applied in practice[46,47,48] and that hybrid development methods are the norm[9,11,10] (cf. Section 4.1), we can pose a number of further questions. From our perspective, the most urgent question is for the actual role of methods, for instance: What is the value of stating that someone, e.g., uses Scrum, when no-one does it by the book? What are the implications for devising a particular method if, in practice, the method starts evolving[16] into a hybrid? What are the implications for software process improvement (SPI) programs if, regardless of the "showroom method" providing the umbrella, the practices are the *stable* key components of organizing and conducting the actual project work? What are the implications for educators if companies require students trained in latest methods[49], but key is a solid understanding of the "old stuff", which is still the core of modern software engineering practice?

Brooks[50] argued that there is no *"Silver Bullet"* that would fit all the different flavors that the software engineering industry has. However, especially in the last decade, it appears that Brooks' observation has been forgotten, and some methods are relentlessly advertised as *the* silver bullet. *New* methods are continuously spawned, and people engage in rather unhealthy discussions arguing whether one method is superior over another. So, are we chasing the white rabbit by focussing on hyped methods and substituting agility with Scrum? One could disagree as "revolutions" like the Agile Manifesto changed the industry inside-out, which is certainly true and industry did progress. Yet, such revolutions changed the mindset and the culture of companies, *not* the practices. Test-driven development, continuous integration, continuous delivery, continuous deployment are incarnations of building blocks that already existed, but, in Beck's words[2] have been "cranked up all the knobs to 10". What has changed is how they are combined and how they are used. We argue that we should distance ourselves from such discussions about the "right" method, but should focus our attention to the practices. Studying the nuances behind practices and their implementation in different contexts would possibly lead to interesting findings whether some hybrid methods (sets of practices) are more effective than others.

As shown in this paper, among practitioners, strong agreement can be found at the practice level, and, when analyzing at this level of granularity, methods and frameworks fade into the background. Therefore, we argue that *researchers* should report on the actual practices when presenting cases, as assumptions on practices used based on a method or framework do not hold, *practitioners* should be mindful about new hypes as the identified core practices are building blocks that are agnostic of methods and, finally, *educators* should put more emphasis on teaching practices rather then methods by explaining the rationale behind them and the different ways in which they can be executed.

## 5.3 | Threats to Validity

In this section, we discuss the threats to validity according to the schema provided by Wohlin et al.[51] and in relation to the constructive validity procedures as outlined in Section 3.4.

**Construct Validity.** First, as we used an online questionnaire, the options provided in the questions might have both been incomplete and prone to misunderstandings resulting in potentially incomplete or wrong answers. Several tactics have been taken to reduce and mitigate these threats. Notably: multiple-choice questions were complemented with a free-text option for potentially necessary clarification; the questionnaire was made available in four different language to lower the risk of misunderstanding due to language issue (translation performed by native speakers); the questionnaire was constructed by a team of researchers, which tested and revised it (see Section 3.2); and, the questionnaire was first released in Europe as a further quality controll[10], which led to final revision. Second, the tactics used to reduce and mitigate the threat of participants not reflecting the target population needs to be detailed, since, following a convenience sampling strategy, links to the survey were spread via multiple networks and mailing lists (Section 3.2.2). The terminology used in the survey required specific knowledge to answer the questionnaire, and the consistently meaningful free-text answers, which have been analyzed qualitatively in Klünder et al.[16], indicate that this threat if present had an insignificant impact. On this regard, it is important to highlight that no definition was provided for any given item, be this a method, a practice, or a technical concept in general. While this was done to avoid any bias that might have been introduced from exposing participants to our knowledge, informing them on the concepts of interest, or leading them towards specific attitudes when answering the survey questions, the lack of definitions also provided a barrier to engage with the survey that reduced the risk of receiving answers from participants outside the target population.

[2] Taken from an interview by informIT, March 23, 2001: http://www.informit.com/articles/article.aspx?p=20972, last access: February 6, 2019.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

**Internal Validity.** Prior to the analysis, we cleaned the data (Section 3.3.1), which could introduce a threat to internal validity as errors might have been introduced. Also, in the data analysis, we did not exclude data points per sé, but performed the analyses with varying $n$'s. To mitigate the risks, all steps have been performed by at least two researchers and have been checked by other researchers not involved in the actual analysis activities. Due to these review processes, we have confidence that the method is reliable and reproducible.

**Conclusion Validity.** The interpretation of the statistical tests is based on a significance level of $p \leq 0.05$, and we found no evidence that allows us to reject our null hypotheses (Table 1). Furthermore, for analyzing sets of methods and practices, we used a 35% and an 85% threshold (Section 3.3.3). Changing these thresholds would influence the results by enlarging the sets of methods and practices. Also, the limited set of options for the multiple-choice questions could influence the findings. The choice of the thresholds can of course be discussed. However, we contend that the effects observed lay the foundation for future research, which is necessary to study the effects in more detail.

**External Validity.** Although our analysis is based on a large dataset (Section 3.3.1), we cannot claim full generalizability. Yet, we reached a broad coverage of domains and participant roles as well as an even distribution of company sizes (Section 4.1). This allows for making observations that are independent of these factors. For other factors, further research is necessary. Nevertheless, the generalization of a single study to all cases of software development is a threat. Moreover, concerning the generalizability of results across countries it would have been necessary to have more data points from Africa, Asia, and North America (Figure 2). Having few data points from countries in these regions threatens the global generalizability of our results. However, the data points that we have, e.g., from Uganda, indicate that our results might be to some degree valid for these regions as well. Future studies are needed to confirm this.

## 6 | CONCLUSION

In this paper, using a large-scale international online survey, we studied the use of *hybrid development methods* in practice. An analysis of 1,467 data points revealed that using different frameworks, methods and practices in combination as hybrid methods is the norm across companies of all sizes and industry sectors. We identified eight base methods and few practices only that find agreement among study participants. For the study participants that explicitly stated to use processes in combination, we could identify 27 base methods and method combinations that, together with three practices forming three pairs, build the basis to devise hybrid methods. We also found that the sets of practices have limited dependencies to the methods. We therefore argue that practices are the building blocks for devising hybrid methods.

In terms of future research, we plan to build on our observations and findings showing that practices are the essential unit of analysis when looking at software development activities within an organization. We note the core set of practices along with the complementary sets of practices identified in Section 4.5 are common to all development methodologies. Because they are so widely deployed, we observe that development organizations see these practices as essential activities enabling them to deliver good software to their customers. We believe that the idea of having a set of common practices that are essential to sound software development has been the motivation behind maturity model frameworks like the CMMI, ISO/IEC 15504 and others. For our future work, we would like to conduct further analysis using the HELENA dataset to explore what having a core set of practices means regarding how industry views the value of maturity model frameworks and specific key process areas within those frameworks. Finally, a future direction of investigation will involve the slicing of the dataset into all its variables to identify whether statistically influencing parameters in the context described by participants can be used to identify recurring inclinations in the choices of hybrid methods. Therefore, answering questions similar to the following: *Does the application domain influence the choice of hybrid methods and how?* or *Does the criticality of the software systems influence the choice of hybrid methods and how?*

## ACKNOWLEDGMENTS

We thank all the study participants and the researchers involved in the HELENA project for their great effort in collecting data. *Dietmar Pfahl* was supported by the institutional research grant PRG887 of the Estonian Research Council as well as the Estonian IT Center of Excellence (EXCITE).

## References

1. Dikert K, Paasivaara M, Lassenius C. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software* 2016; 119: 87 - 108. doi: https://doi.org/10.1016/j.jss.2016.06.013

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

2. Dingsøyr T, Nerur S, Balijepally V, Moe NB. A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software* 2012; 85(6): 1213–1221.

3. Fitzgerald B, Hartnett G. A Study of the Use of Agile Methods within Intel. *IFIP International Working Conference on Business Agility and Information Technology Diffusion* 2005; 180: 187–202.

4. Garousi V, Coşkunçay A, Betin-Can A, Demirörs O. A survey of software engineering practices in Turkey. *Journal of Systems and Software* 2015; 108: 148 - 177. doi: https://doi.org/10.1016/j.jss.2015.06.036

5. Laanti M, Salo O, Abrahamsson P. Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation. *Inf. Softw. Technol.* 2011; 53(3): 276 - 290. doi: https://doi.org/10.1016/j.infsof.2010.11.010

6. Murphy B, Bird C, Zimmermann T, Williams L, Nagappan N, Begel A. Have Agile Techniques been the Silver Bullet for Software Development at Microsoft. *ACM / IEEE International Symposium on Empirical Software Engineering and Measurement* 2013: 75–84.

7. Diebold P, Ostberg JP, Wagner S, Zendler U. What do practitioners vary in using scrum?. *International Conference on Agile Software Development* 2015: 40–51.

8. Reddy A. *The Scrumban [r] evolution: getting the most out of Agile, Scrum, and lean Kanban*. Addison-Wesley Professional. 2015.

9. West D, Gilpin M, Grant T, Anderson A. Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today. tech. rep., Forrester Research Inc.; 2011.

10. Kuhrmann M, Diebold P, Münch J, et al. Hybrid Software and System Development in Practice: Waterfall, Scrum, and Beyond. *International Conference on Software and System Process* 2017: 30–39. doi: 10.1145/3084100.3084104

11. Theocharis G, Kuhrmann M, Münch J, Diebold P. Is Water-Scrum-Fall Reality? On the Use of Agile and Traditional Development Practices. *International Conference on Product Focused Software Development and Process Improvement* 2015: 149–166.

12. Vijayasarathy LR, Butler CW. Choice of Software Development Methodologies: Do Organizational, Project, and Team Characteristics Matter?. *IEEE Software* 2016; 33(5): 86–94. doi: 10.1109/MS.2015.26

13. Meier A, Kropp M. Swiss Agile Study. Online http://www.swissagilestudy.ch; 2019.

14. VersionOne . State of Agile Survey. Available from: http://www.versionone.com/agile-resources/more-resources/blogs/; 2006-2014.

15. Kuhrmann M, Diebold P, Münch J, et al. Hybrid Software Development Approaches in Practice: A European Perspective. *IEEE Software* 2019; 36(4): 20++31.

16. Klünder J, Hebig R, Tell P, et al. Catching up with Method and Process Practice: An Industry-Informed Baseline for Researchers. *Proceedings of the International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)* 2019: 255–264.

17. Kuhrmann M, Tell P, Klünder J, Hebig R, Licorish SA, MacDonell SG. Complementing Materials for the HELENA Study (Stage 2). [online] DOI: 10.13140/RG.2.2.11032.65288; 2018.

18. Tell P, Klünder J, Küpper S, et al. What Are Hybrid Development Methods Made of? An Evidence-based Characterization. *Proceedings of the International Conference on Software and System Processes* 2019: 105–114. doi: 10.1109/ICSSP.2019.00022

19. Royce WW. Managing the Development of Large Software Systems. *Proceedings of IEEE WESCON* 1970: 1–9.

20. Boehm BW. A spiral model of software development and enhancement. *Computer* 1988; 21(5): 61–72.

21. GPM Dt. Gesellschaft f. Projektmanagement. *Status Quo Agile*. GPM. 2017.

22. Tripp JF, Armstrong DJ. Exploring the relationship between organizational adoption motives and the tailoring of agile methods. *47th Hawaii International Conference on System Sciences (HICSS)* 2014: 4799–4806.

23. Begel A, Nagappan N. Usage and Perceptions of Agile Software Development in an Industrial Context: An and Exploratory Study. *International Symposium on Empirical Software Engineering and Measurement* 2007: 255-264.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

24. Nerur S, Mahapatra R, Mangalaraj G. Challenges of Migrating to Agile Methodologies. *Communications of the ACM* 2005; 48(5): 73–78.

25. Van Waardenburg G. VVH. When agile meets the enterprise. *Information and Software Technology* 2013; 55(12): 2154–2171.

26. Cusumano M, MacCormack A, Kemmer CF, Crandall B. Software Development Worldwide: The State of the Practice. *IEEE Software* 2003; 20(6): 28–34.

27. Jones C. Variations in Software Development Practices. *IEEE Software* 2003; 20(6): 22–27.

28. Neill CJ, Laplante PA. Requirements engineering: the state of the practice. *IEEE Software* 2003; 20(6): 40-45. doi: 10.1109/MS.2003.1241365

29. Cockburn A. Selecting a Project's Methodology. *IEEE Software* 2000; 17(4): 64–71.

30. Boehm B, Turner R. Using Risk to Balance Agile and Plan-Driven Methods. *IEEE Computer* 2003; 36(6): 57–66.

31. Clarke P, O'Connor RV. The Situational Factors That Affect the Software Development Process: Towards a Comprehensive Reference Framework. *Inf. Softw. Technol.* 2012; 54(5): 433–447.

32. Kalus G, Kuhrmann M. Criteria for Software Process Tailoring: A Systematic Review. *International Conference on Software and Systems Process* 2013: 171–180.

33. Rombach D. Integrated Software Process and Product Lines. *International Software Process Workshop* 2005: 83–90.

34. Kuhrmann M, Ternité T, Friedrich J, Rausch A, Broy M. Flexible Software Process Lines in Practice. *Journal of Systems and Software* 2016; 121(C): 49–71. doi: 10.1016/j.jss.2016.07.031

35. OMG. Essence – Kernel and Language for Software Engineering Methods. OMG Standard formal/18-10-02, Object Management Group; 2018.

36. Jacobson I, Lawson H, Ng PW, McMahon PE, Goedicke M. *The Essentials of Modern Software Engineering: Free the Practices from the Method Prisons!* ACM BooksMorgan & Claypool Publishers. 2019.

37. Jacobson I, Ng PW, McMahon PE, Spence I, Lidman S. *The Essence of Software Engineering: Applying the SEMAT Kernel*. Addison-Wesley Professional. 2013.

38. Joint Technical Committee ISO/IEC JTC 1, Subcommittee SC 7 . Software Engineering – Metamodel for Development Methodologies. Tech. Rep. ISO/IEC 24744:2007, International Organization for Standardization; 2007.

39. OMG. Software & Systems Process Engineering Metamodel Specification (SPEM) Version 2.0. tech. rep., Object Management Group; 2008.

40. Kuhrmann M, Fernández DM, Steenweg R. Systematic Software Process Development: Where Do We Stand Today?. *Proceedings of the International Conference on Software and Systems Process* 2013: 166–170.

41. Solinski A, Petersen K. Prioritizing agile benefits and limitations in relation to practice usage. *Software Quality Journal* 2016; 24(2): 447–482.

42. Cusumano M, MacCormack A, Kemerer CF, Crandall B. Software development worldwide: the state of the practice. *IEEE Software* 2003; 20(6): 28-34. doi: 10.1109/MS.2003.1241363

43. Robson C, McCartan K. *Real World Research*. John Wiley & Sons. 2016.

44. Rupert Jr G, others. *Simultaneous statistical inference*. Springer Science & Business Media. 2012.

45. Diebold P, Zehler T. The Right Degree of Agility in Rich Processes. *Managing Software Process Evolution: Traditional, Agile and Beyond – How to Handle Process Change* 2016: 15–37.

46. Mathiassen L, Munk-Madsen A, Nielsen PA, Stage J. Method Engineering: Who's the Customer?. *Method Engineering: Principles of method construction and tool support* 1996: 232–245.

47. Fitzgerald B. Formalized systems development methodologies: a critical perspective. *Information Systems Journal* 1996; 6(1): 3-23.

48. Dittrich Y. What does it mean to use a method? Towards a practice theory for software engineering. *Information and Software Technology* 2016; 70: 220 - 231.

©Wiley. PREPRINT. This is the author's version of the work. Not for redistribution.

49. Kuhrmann M, Nakatumba-Nabende J, Pfeiffer RH, et al. Walking through the Method Zoo: Does Higher Education really meet Software Industry Demands?. *Proceedings of the International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)* 2019: 1–11.

50. Brooks FP. No silver bullet. *IEEE Computer* 1987; 20(4): 10–19.

51. Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B, Wesslén A. *Experimentation in software engineering*. Springer Science & Business Media. 2012.

□

View publication stats