Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

On the benefits and problems related to using Definition of Done — A survey study[☆]

Sylwia Kopczyńska[*], Mirosław Ochodek, Jakub Piechowiak, Jerzy Nawrocki

Poznan University of Technology, Institute of Computing Science, ul. Piotrowo 2, 60-965 Poznań, Poland

ARTICLE INFO

Article history:
Received 10 September 2021
Received in revised form 5 July 2022
Accepted 3 August 2022
Available online 6 August 2022

Keywords:
Definition of Done
DoD
Agile
Scrum
Survey

ABSTRACT

Context: Definition of Done (DoD) is one of the fundamental concepts of Scrum. It expresses a shared view of a Scrum Team on what makes an increment of their product complete. DoDs are often defined as checklists with items being requirements towards software (e.g., quality requirements) or towards activities performed to make the increment shippable (e.g., code reviews, testing). Unfortunately, the knowledge about the usefulness of DoD is still very limited.
Objective: The goal is to study what benefits using the DoD practice can bring to an agile project, what problems it may trigger, and how it is created and maintained.
Methods: In the survey among members of agile software development projects, 137 practitioners from all over the globe shared their experience with us.
Results: 93% of the respondents perceive DoD as at least valuable for their ventures. It helps them to make work items complete, assure product quality, and ensure the needed activities are executed. However, they indicated that every second project struggles with infeasible, incorrect, unavailable, or creeping DoD.
Conclusions: It follows from the study that DoD is important but not easy to use and more empirical studies are needed to identify best practices in this area.

1. Introduction

Organizations are increasingly adopting agile methods. From the results of the PMI survey, it follows that over 70% of organizations report using agile methods (Project Management Institute, 2017), and Scrum is one of the most popular among them.

In Scrum, Developers are to deliver increment(s) of "done" product in each Sprint. It means that an increment must be complete and in usable condition (often referred to as working software) and meet the agreed Definition of Done (DoD). According to the Scrum Guide (Schwaber and Sutherland, 2020) "*the Definition of Done is a formal description of the state of the Increment when it meets the quality measures required for the product.*" DoD might be standardized at the level of organization, or could be created separately for each product. Also, it usually evolves in time to include more stringent criteria as the team matures. Although the use of DoDs is characteristic to the Scrum framework, it is also recommended by other agile/lean frameworks like The Kanban Method (Theobald et al., 2019), and appears in several Agile Maturity Models (AMMs) (Nurdiani et al., 2019).

The most recent version of Scrum Guide (Schwaber and Sutherland, 2020) states that a DoD "*creates transparency by providing everyone a shared understanding of what work was completed as part of the Increment.*" Therefore, it is essential that a DoD is formulated in such a way that all team members understand it in the same way. Although there are no official guidelines on how to achieve this, most of the teams end up defining their DoDs as lists of "items", which need to be satisfied to claim that a given Sprint is "done". DoD items concern, e.g., source code quality, performing code review, testing activities, completing work items, or non-functional requirements regarding cross-cutting concerns, like security or performance (Silva et al., 2017; Alsaqaf et al., 2019).

The existing body of knowledge regarding the practical usage of DoD is very limited (Silva et al., 2017). Taking into account little awareness of the importance of working towards the shared understanding of what "done" means in some teams (O'Connor, 2010; Paasivaara et al., 2018), it would be especially valuable to study the potential benefits and problems related to using DoD to bring new arguments to the discussion about how much we should invest in the preparation of DoDs.

The goal of this paper is to investigate how practitioners use Definition of Done (DoD) in software projects and product teams.

[☆] Editor: Matthias Galster.
[*] Corresponding author.
E-mail addresses: sylwia.kopczynska@cs.put.poznan.pl (S. Kopczyńska), miroslaw.ochodek@cs.put.poznan.pl (M. Ochodek), jakub.k.piechowiak@gmail.com (J. Piechowiak), jerzy.nawrocki@cs.put.poznan.pl (J. Nawrocki).

https://doi.org/10.1016/j.jss.2022.111479
0164-1212/© 2022 Elsevier Inc. All rights reserved.

In the further paragraphs, for simplicity, we will refer to both software projects and product teams as *projects*.[1] The special focus of the paper is on the usefulness and the problems DoD might trigger. To achieve the goal, we conducted a survey among practitioners from all over the world. The main contributions of our study are as follows:

- we investigated to what extent the practitioners perceive the DoD practice as valuable,
- we studied 19 (and identified two more) benefits of using DoDs by investigating their frequency of appearance in agile projects,
- we identified 19 problems that are triggered by lack of using DoD in a project,
- we studied the frequency and significance of 19 problems that agile teams might encounter while using DoDs,
- finally, we investigated what DoD items are, and the process of creating and maintaining DoDs, focusing on the steps of that process, the roles that are involved, and the tools that are used.

This paper is organized as follows. First, we explain what DoD is in Section 2. Then, in Section 3, we discuss the related studies. Next, in Section 4, we describe the design of the survey and discuss the validity threats. The demographic information about our respondents is presented in Section 5, while the results of the survey are presented in Section 6. The implications of our study are discussed in Section 7. Section 8 concludes our findings.

2. Background

According to Scrum Guide (Schwaber and Sutherland, 2020), each Sprint is to deliver an Increment of a potentially releasable product which is usable and adheres to the Scrum Team's "Definition of Done". However, the guide does not provide specific guidelines on how DoD should look like. However, it draws attention to its high value, i.e., complying with DoD is required at the end of each Sprint to ensure transparency and assure product quality. Practitioners describe DoD as a "social contract in agile teams" that "acts as a check before work is allowed to leave the Sprint" (Power, 2014a).

However, the concept of DoD is not only used in Scrum but also in other agile approaches. For example, The Kanban Method recommends to define the workflow, that is the specific process each item-to-do needs to undergo with the defined steps (stages) and the policies specifying the requirements that need to be satisfied to let the item flow from one step to another. One type of such policies is DoD (Anderson, 2016).

In practice, DoD frequently has a form of a checklist with requirements that need to be assessed to determine whether the work is "done", see examples in Fig. 1 (the examples ①-⑩ are taken from the DoD items the respondents shared with us in the survey). DoD items mainly focus on Silva et al. (2017), Madan (2019), Kopczyńska et al. (2020), Saddington (2012) and Alsaqaf et al. (2019): (1) business or functional requirements, i.e., require certain requirement(s) to be implemented or tested or verified, e.g., ② ; (2) quality aspects, e.g., concern the status of testing, the level of technical debt, the results of code reviews such as ③, ④, ⑤, ⑥, and (3) non-functional requirements, i.e., they state if the desired characteristics of a system are met, e.g., ⑨. They might concern different types of entities that appear in software development (e.g., test (③), code (⑤), defects (④), documentation (⑨)) and be defined at multiple "levels", e.g., for a single task, for a user story/feature (①), for all user stories/features (②), for a whole increment (⑦), for a release (⑧).

[1] We investigate both software development initiatives planned for a certain period of time (projects) as well as those continuously developing some products (product teams).

① Feature merged to develop branch
② All features/tasks completed
③ All tests pass
④ No open defects
⑤ No TODOs in code
⑥ For coding/development tickets you must have satisfied Code Coverage
⑦ Build passes
⑧ Deployed to Production
⑨ Functional and Technical Specifications completed
⑩ All documentation updated

Fig. 1. Excerpts from DoDs presenting different types of DoD items.

DoD items are sometimes confused with Acceptance Criteria (AC). However, those two have slightly different goals. AC is a set of requirements that must be fulfilled for a given Product Backlog Item, so it can be accepted by Product Owner, customer, user, or other stakeholder (Rubin, 2013). Testing against AC produces a clear pass/fail result if the product can be accepted or not (Povilaitis, 2014). Thus, ACs are item-specific and orthogonal to DoD. Viscardi (2013) states that DoD is a quality goal, while AC are functional or behavioral expectations. AC are not directly mentioned in the Scrum Guide, however they are an important and useful tool in agile projects. Example ACs for an e-commerce site, which follow from the experience of the authors, could be AC1:"*Lists 1.000 products on a one page*" or AC2: "*Sorts the listed products with respect to price and name in both ascending and descending order*" while DoD items could be DoDI1: "*Deployed to the production environment*", DoDI2: "*Each feature was code reviewed by at least one dev who is not the author and the improvements are introduced*".

While the notations of user stories or use cases are frequently used to document software requirements (Cohn, 2010), there is no common format in which DoD items are specified. Frequently, DoD items are short statements or short sentences such as those presented in Fig. 1 or mentioned in the work of Silva et al. (2018).

3. Related work

The usage of DoD and its value were discussed in several papers describing lessons learned from industry projects. For example, according to Davis (2013), using DoD has a positive impact on reducing the number of defects and limiting technical debt. Power (2014b) claims that DoDs help with assessing the completion of work. Taipale (2010) found that their software development with DoD results in the product of high quality and is a well-managed development process. Masood et al. (2020) observed that a well-defined DoD supports self-assignment of tasks in agile teams. Kasauli et al. (2020) proposed using DoD items at the level of user stories to foster requirements traceability in a project. Several other agile practitioners claim that DoD is an essential practice and recommend using it, e.g., Saddington (2012), Jakobsen and Johnson (2008) and Cohn (2010). Although these papers mention some benefits of using DoD, they base their observations either on the authors' experience or on analyses of individual cases. Our study complements the previous findings

Table 1
Comparison of related work concerning experience in agile software development that revealed benefits or problems concerning the usage of DoD.

	Object of study	Type of study	Participants	Domain	Problems or Benefits
Power (2014a)	Using Definition of Ready	Case study	One company, overall opinion	Networking	Identified a benefit
Taipale (2010)	Process used in Lean startup	Case study	One company, overall opinion	Software development, retail	Identified a benefit
Masood et al. (2020)	Self-assignment to work	Grounded theory	23 companies, 42 practitioners	Software development	Identified a benefit
Kasauli et al. (2020)	Challenges and strategies using agile approaches	Case study	One company, two departments	Automotive	Identified a benefit
Saddington (2012)	Scaling product ownership	Case study	A program by the Department of Defense and US Air Force, 4 teams	Military	Identified a benefit
Jakobsen and Johnson (2008)	Combining CMMI with Scrum	Case study	One company	Software and systems company	Identified a benefit
Cohn (2010)	Theory and experience about agile	No study, book	Own experience	–	Share possible benefits
O'Connor (2010)	Agile transformation	Experience description	One company, one project	Education	Identified a problem
Igaki et al. (2014)	Ticket driven development for teaching Scrum	Experiment with students	One company, one project	Education	Identified a problem
Alsaqaf et al. (2019)	Way developers treat quality requirements in agile	Multi-case study, interviews	17 agile practitioners from 6 organizations	Public sector, government, commercial, banking, commercial navigation, health care, insurance, telecom	Identified a problem
Our study	Practice of DoD	Survey	137 agile practitioners	Over 20 different domains	Analyzed problems and benefits and the process of using DoD

by identifying and helping to understand the benefits of using DoDs in agile projects. We also scale up the existing research by surveying a large sample of projects.

While the majority of authors discuss the advantages of creating and using DoD, only a few of them mention the problems that accompany that process. For instance, O'Connor (2010) states that there might be some conflicts between business and developers on what "done" means, while Igaki et al. (2014) emphasize the need of making sure that the DoD items are followed. Igaki et al. (2014) and Alsaqaf et al. (2019) found that DoD might become too extensive and as a result negatively impact team velocity or make validation more complex.

Although these papers present valuable lessons learned from implementing and teaching agile methods, neither of them provides a comprehensive and well-evaluated set of problems related to using DoDs. With our study, we fill this knowledge gap. We use the identified problems and benefits as the basis, extend them with those from our knowledge and experience, and evaluate their importance among practitioners all over the world. We compared the related works mentioned in two previous paragraphs with each other and with our work in Table 1.

There are two works in which the focus was given to DoD. First, Silva et al. (2017) conducted a Systematic Literature Review on the papers published prior to 2017. The review aimed at investigating what the done criteria are, what is the context of the teams using DoD, and what the characteristics of the studies investigating the topic of DoD are. They found 8 relevant studies and analyzed them. As a result, they characterized the items that are commonly present in DoDs. For instance, they observed that DoD items are often defined at four levels: story, Sprint, release, and project. They found that four primary studies mention the benefits of using DoD. Finally, based on the analysis of gaps in the literature regarding DoD, they concluded that "*there is a need for more and better empirical studies documenting and evaluating the use of the DoD in agile software development*". In their follow-up study, Silva et al. (2018) conducted a survey on a sample of 20 respondents. The study provided some insights on how DoD is created and maintained, e.g., they learned that 80% of respondents had their DoDs explicitly documented, the DoDs were mostly created by the team and evolved throughout the project. Our study scales-up and complements both of these studies, providing some new insights on DoD items, on how DoDs are created and maintained, and investigates the importance of the identified benefits and problems. The comparison of the studies with each other and with our study is presented in Table 2.

4. Research methodology

4.1. Research aims and questions

Our goal is to study the usefulness of DoDs from two perspectives. Firstly, we are interested in learning what benefits a DoD might bring to a project or product team.[2] Secondly, we want to identify and evaluate the problems associated with implementing the DoD practice. Finally, we would like to investigate whether there are any relationships between the problems encountered while using DoD and the benefits of using it. The existence of such relationships would mean that when certain problems with DoD exist, some benefits are less/more likely to be observed. Thus, it might help to make decisions about problem-mitigation strategies.

Based on these three goals, we formulate the following research questions:

- RQ1. *What are the benefits of using DoD?*
- RQ2. *(a) What are the problems encountered while using DoD and (b) how are they mitigated?*

2 Scrum and other agile approaches can be used both in software projects and in product teams. For simplicity, we will refer to both as *projects* in this paper.

Table 2
Comparison of related work concerning the practice of using DoD with each other and with our work.

Type of study	Silva et al. (2017) SLR	Silva et al. (2018) Survey	Our work Survey
Research questions	1. What are done criteria? 2. In which environment DoD was used? 3. What types of study tackle DoD?	1. What done criteria are? 2. What process is used to define DoD? 3. What levels of DoD are used? 4. Is DoD emergent during a project? 5. How is DoD established and maintained? 5. Are DoDs explicitly documented? 6. How are DoD criteria assessed? 7. Effectiveness of DoD 8. Challenges in using DoD	1. What are the benefits of using DoD? 2. What are the problems encountered while using DoD and how they are mitigated? 3. Problems–Benefits relationships 4. What DoD items are?
Subjects/ Participants	8 primary studies	20 agile practitioners from 16 countries	137 agile practitioners from 45 countries
#DoD items analyzed	62	22	143
Problems or Benefits	Found the benefits mentioned in four primary studies	Asked respondents about benefits and challenges	Used the identified benefits and problems to survey practitioners from all over the world about perceived importance and identify any new items.
Practice of using DoD	–	Asked with open questions respondents about the definition, evolution, assessment, and if DoD is documented	Asked with open questions about the whole process of using and maintaining DoD, person responsible, and format.
DoD item analysis	Categorized DoD items and investigated the frequency	Listed the DoD items using the categorization previously proposed	Analyzed DoDs using the previously proposed categorization

- RQ3. *Are there any relationships between the problems and benefits while using DoD?*

As follows from the literature review presented in Section 3, the body of knowledge regarding the process of creating and maintaining DoDs is very limited. Therefore, we formulate one more research question to broaden our understanding of that process:

- RQ4. *How the DoD practice is implemented?*
  - *(a) How DoD is created and maintained?*
  - *(b) What are the DoD items?*

Additionally, answering this question would allow us to characterize the context in which the DoDs were created and used. Consequently, it might help us understand the origins of the benefits and problems.

4.2. Research method

We chose questionnaire-based Survey Research as our research method of choice. Since this method allows for collecting and analyzing large samples of projects in a cost-effective way, it gives us the possibility to draw an overall picture of what benefits and problems one might expect to encounter when adopting the DoD practice in a project. While designing our study, we followed the guidelines provided by Wohlin et al. (2012) and by Molléri et al. (2020). Our survey can be classified as a descriptive study, however, since the topic we investigated has not been deeply studied, it could also, to some degree, be treated as an exploratory study.

The questionnaire is available as supplemental material for this paper.

4.3. Population and sample representatives

Our target population constitutes participants of agile software development projects and product teams that use DoD practice. We assume that a representative sample of the target population shall have the following characteristics:

1. Using the DoD practice — the sample includes responses from the participants that used the DoD practice in the projects they refer to.
2. Year — projects being referred to are contemporary agile projects. Referring to old projects would constitute a serious threat to internal and external validity since the usage of agile methods and their popularity has changed visibly over the last years (Hoda et al., 2018).
3. Context (country, domain, etc.) — the distributions of different project context factors in the sample of projects should be similar to the corresponding distributions in the IT industry.
4. Agile method — we assume that the DoD practice is mainly used by the teams working according to the Scrum guidelines (since the DoD practice originates from Scrum). Therefore, we assume that our target population is dominated by Scrum teams.

We are going to evaluate the representatives of the sample based on the above given characteristics. The first two characteristics (1 and 2) are filtering criteria that should be included in the survey instrument. The two remaining characteristics (3 and 4) are more difficult to evaluate since we do not have accurate and precise information on how often given context factors appear in IT projects. Therefore, we are going to base our evaluation on the available data from surveys concerning agile methods (Digital, 2021; Ochodek and Kopczyńska, 2018; Matharu et al., 2015) and the characteristics of the large sample of IT projects collected in the ISBSG database and reported by Hill (2011). In particular, we are going to take into account the geographic dispersion of the respondents/projects and the agile methods that are reported in the surveys, while the ISBSG database allow us to assess the representatives of different domains and types of applications in the sample.

4.4. Survey instrument

We designed an online questionnaire divided into four parts and implemented it using the Survey Monkey platform.

The first part consisted of three pages: a welcome page (presenting the goal of the study and providing contact information to the researchers conducting the study), a page providing some basic information about the concept of DoD (as one of the means to ensure construct validity), and a page asking the respondents to focus on one of the projects or product teams they participated in which a DoD was used. We based the description of DoD on the Scrum Guide (Schwaber and Sutherland, 2020). Also, we asked about the date of the last participation of the respondent in a project in which a DoD was used. It had two purposes: (1) we wanted each respondent to focus on an individual project, and (2) we wanted to verify whether the responses regard new developments.

The second part of the questionnaire regarded the value of using a DoD in a project. First, we asked the respondents to evaluate how valuable was using a DoD in their project (in general) by using a five-point Likert scale. Second, there was a list of 19 potential benefits to be assessed individually. The list of benefits resulted from our literature review and was extended by a few potential benefits that we brainstormed. Also, the respondents could select the option "other" to provide their own examples of benefits. For each benefit, the respondent was asked if the benefit to their project is the result of using a DoD. They responded using a five-point ordinal scale ("Definitely YES", "YES", "Neither YES nor NO", "NO", "Definitely NO"). We also allowed for the answer "I don't know" to avoid introducing bias by forcing respondents to provide answers (e.g., if the respondents lacked the knowledge to answer the question reliably or did not want to express their opinion).

The third part of the questionnaire contained a series of questions concerning problems with using DoDs. The first two questions asked the respondent to assess the negative impact of the possible problems in their projects concerning management and DoD items, respectively. The list of problems had two sources. We had extracted the problems mentioned in the literature. Next, since DoD items are requirements, we extended the set with the common problems by specifying requirements from Wiegers and Beatty (2013). For each problem, the respondents could select one of the following answers: "Problem did not appear in the project", "Problem had a positive impact" (e.g., there was a problem with using DoD but its presence triggered some corrective actions that were beneficial for the project) or assess the significance of the negative impact of the problem if it had appeared using a five-point Likert scale ("Definitely significant", "Rather significant", "Neither significant nor NOT significant", "Rather NOT significant", "Definitely NOT significant"). Again, we also allowed for the answers "I don't know" and "Other". Finally, we asked the respondents how the problems with using a DoD were solved (an open question).

In the fourth part, we placed open questions concerning the process of creating and maintaining DoDs (the process, the roles involved, the frequency of updating the DoD, and the ways of publishing it).

The fifth part contained 11 questions asking about demographic information. We used them to characterize the sample of respondents. In particular, we asked about the experience, domains, type of applications, methods, size of the project, and size of the organization. The answers to the questions could be provided using ordinal scales or as multiple-response true/false questions (more than one choice was allowed). For the questions concerning the domains and application types, we had adopted the classification scheme used by ISBSG (2020), while for the size of the organization, we used the classification defined by the European Commision (2020).

The last part of the questionnaire asked the respondents to leave any comments, questions, or remarks. We also asked them to voluntarily share their DoD and the problems they met in some other project caused by the lack of DoD. Optionally, every respondent might have provided their email address to get a summary of the results after the survey is completed.

4.5. Survey instrument validation and evolution

The prepared questionnaire underwent multiple internal and external reviews. First, it was examined by the authors of this paper and the initial version was subjected to a pilot study with 7 participants (four members of our research group and three external agile software development professionals). Their feedback allowed us to improve the wording of some questions, and a few spelling mistakes were found. One wording issue concerned that we asked about "software projects" and two practitioners had doubts if they are eligible to participate in the survey. They pointed out that agile approaches are also used in the context where there is no defined period of time to finish the work (projects) and they are usually called "product teams". Next, we found out that many software engineers have the same opinion, e.g., Sriram Narayan or Martin Fowler (Narayan, 2018). Thus, we added the information to the questionnaire. However, no major problems were identified. The questionnaire was not further modified during the study.

4.6. Ethical considerations

While designing our study, we have considered a series of ethical considerations, especially those discussed by Vinson and Singer (2008).

Informed consent. Participation in the study was voluntary. The invitation letter and the introduction page of the questionnaire form informed potential participants about the goal of the study, the research group conducting the study, the research procedure (including information on how the responses will be processed, and the results communicated), the benefits of participating, the estimated time to complete the survey, and the contact e-mail addresses of the members of the research team.

Anonymity. Participation in the study was anonymous. We did not ask about any personal information or the names of the companies. However, the participants could provide us with their e-mail addresses that might contain their names or surnames. Therefore, we excluded these data from further analyses.

Beneficence. A direct benefit of participating in our study was the early access to the survey results before they are officially published.

Confidentiality. The online survey was conducted using the Survey Monkey platform, which we consider to be a secure service. From our analysis of the security features, it follows that they are ISO 27001 certified, EU and US Privacy Shield Certified. They use AES 256 based encryption and declare to implement appropriate technical, organizational, and administrative systems, policies, and procedures to ensure the security, integrity, and confidentiality of the questionnaire and collected data to mitigate the risk of unauthorized access to or usage. The only sensitive data concerned e-mail addressees of the respondents. They were processed separately from the remaining data and used only to send the results back to the respondents.

4.7. Data collection

Since there is no one 'place' that provides access to the representatives of the population, we decided to target the respondents using Internet-based channels. We decided to conduct an invitation-based online survey, and, thus, we sent messages to people we knew to have experience in Agile and posted a request

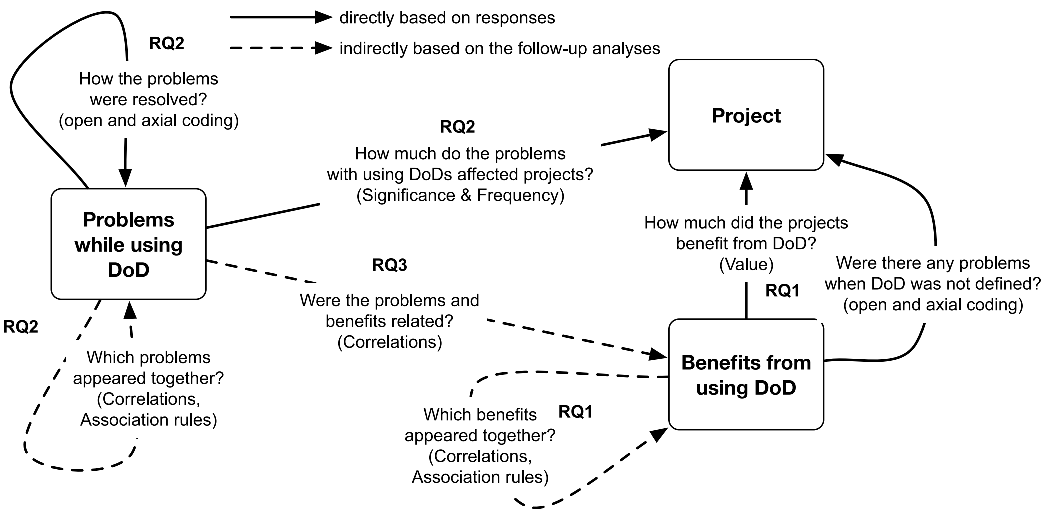

directly based on responses
indirectly based on the follow-up analyses
RQ2
How the problems were resolved? (open and axial coding)
Problems while using DoD
RQ2
How much do the problems with using DoDs affected projects? (Significance & Frequency)
Project
How much did the projects benefit from DoD? (Value)
RQ1
Were there any problems when DoD was not defined? (open and axial coding)
RQ3
Were the problems and benefits related? (Correlations)
Benefits from using DoD
RQ2
Which problems appeared together? (Correlations, Association rules)
Which benefits appeared together? (Correlations, Association rules)
RQ1

Fig. 2. Analyses performed and methods used to answer RQ1, RQ2, and RQ3.

to participate in the survey in social network groups related to Agile on LinkedIn, Facebook, and MeetUp. After three to four weeks, we sent kind reminders to the groups. The data collection took place between February 21, 2020 and September 21, 2020. The exact response rate cannot be calculated due to the usage of public invitations, but given that we collected 276 responses, the response rate can be interpreted to be low, which is characteristic to online surveys.

There were 137 complete and 139 incomplete responses. The respondents spent on average ca. 17 min. (median) to complete the survey (the maximum duration was 72 h 31 min. while the minimum duration was 4 min.). There were three subgroups among those who have not completed the survey (we will refer to them as incomplete respondents). The majority of incomplete respondents 69% (96) answered just the first question, that is about the year of their project and devoted to the survey ca. 1 min. (median). Another 26% (36) of incomplete respondents spent ca. 4 min. (median) and answered the next question—the first question that assessed the benefits of DoD. Finally, 5% (7) of incomplete respondents dedicated ca. 7 min. to the survey and left it after sharing their opinion about the problems concerning using DoD. Thus, it seems that the major issues that could discourage our respondents were either (1) the topic of the survey that after reading the introduction was not compelling enough to continue, or they did not have experience in the area, (2) the lists of benefits gave not a good impression, e.g., seemed time-consuming or complicated, or (3) answering the first three groups of questions (year, benefits, problems) made respondents not willing to continue, e.g., tired or bored.

4.8. Data analysis methods

We used the following criteria to validate responses: (1) a respondent has to answer all obligatory questions (to eliminate drop-outs), (2) the answers to all open questions are relevant (to eliminate misleading data), and (3) the last participation in the project the respondent focuses on in the survey was no earlier than 2015. This validation approach resulted in having no missing data and allowed us to apply quantitative analysis methods. Fig. 2 presents a map of analyses performed to answer RQ1–RQ3 and the analysis methods employed for that purpose.

We used frequency analysis to analyze the direct responses to the multi-choice questions. For the questions related to problems and benefits, we introduced additional measures by normalizing the number of responses with respect to the total number of responses (we introduce each of these measures in Section 6 while discussing the results). Also, we used correlation analysis and association rules to analyze the relationships between benefits (RQ1), problems (RQ2), and both of them (RQ3). In particular, we decided to base the correlation analysis on the Spearman rank-order correlation coefficient (Spearman's ρ) since the responses were expressed on ordinal scales. We employed statistical inference testing to filter the relationships that are unlikely to be observed by chance only (we set the significance level $\alpha = 0.05$) and followed the guidelines for interpreting the effect size of relationships provided by Akoglu (2018). Finally, we augmented the correlation analysis by mining association rules with the use of the Apriori algorithm (Agrawal et al., 1994) as a tool for identifying new or confirming the previously identified multi-factor relationships between the problems and benefits. When mining association rules, we converted the ordinal response scales to dichotomous scales to express whether a given benefit/problem was present or absent in a project (positive responses, e.g., "rather significant" and "definitely significant", were mapped to "one" while remaining scale items were replaced with "zero"). We set the minimum support to 0.1 (support is an indication of how frequently the itemset appears in the dataset) and confidence to 0.9 (confidence is the likelihood that item Y is present if item X is present). Unfortunately, the number of inferred rules might be too large to allow for their direct analysis. Therefore, we choose correlation analysis as our main tool for performing the analyses and consider association rules as a supporting one. Finally, although we have to emphasize that identifying a relationship (correlation) between factors does not indicate the existence of causality between them, we attempted to hypothesize about the possible reasons behind the observed relationships.

For the open-text questions we used the grounded theory techniques of coding (open and axial coding) and constant comparison as recommended by Charmaz (2006). To assure reliability of the coding process, we executed the following steps.
Step 1: Preparation—one of the authors (the 3rd author) extracted all responses to open-text questions and moved them into separate files (one file per each question).
Step 2: Individual coding–two researchers (the 1st and the 3rd author) performed (individually) open coding of the text responses, and each code summarized a single key concept. For example in the answer of respondent R15 to the question Q10 about the process of creating DoD: " *Initially we had a generic template. Then, over time we discussed each point and adjusted it to our*

situation. We also discussed whether there is something missing and added our own points.", the codes of `Team` and `Initial template` were identified. Step 3: Creating the coding schema—the two researchers compared their coding schemes and discussed their findings. They applied constant comparison to group similar codes. As a result, they created the final coding schema and formulated guidelines on how to interpret and analyze the quotes. In the example from Step 2, the coding schema of categories and subcategories was developed containing:
• `Who creates DoD?` *(Category)*
→ `Team, Product Owner, Scrum Master` *(SubCateg.)*
• `What is used to create DoD?`
→ `Initial template, Standard of organization`
• `What activities?`
→ `Adjust`
etc.
Step 4: Coding confirmation and final coding—one researcher (the 1st researcher) performed coding using the final coding scheme and the guidelines.
Step 5: Validation of the coding schema—yet another author of the paper (the 2nd author) went through the data and the coding schema, which resulted in clarification of three codes descriptions and not finding any coding errors.
Step 6: Identification of the relationships between codes—two researchers (the 1st and the 2nd author) were identifying relationships between and within codes. For example, we identified several activities concerning using DoD in all answers to the question 10 mentioned in Steps 2&3 but also in other open text questions, such as `Adjust template`, `Discuss`, `Document`, `Analyze`. In this step number 6 we analyzed them to investigate which one follows or proceeds others, which shall be treated as obligatory, and discover any conditions under which the relationships hold. Although, the step was applied to all open questions, it turned out to be the most valuable for question Q10 concerning the process of using DoD.

To analyze the responses to the open question asking to voluntarily provide the DoD, we used the following four step approach:
Step 1: Preparation—the first author of the study extracted DoD items from the answers and placed them in a table in an Excel spreadsheet.
Step 2: Categorization—the first author assigned the category proposed by Silva et al. in Silva et al. (2017, 2018) to each DoD item. The second author went through the assigned items and verified the correctness of assignment. After verification, during a meeting 4 changes were introduced.
Step 3: Subject identification—the first author extracted subjects from each DoD item, next, the second author went through the subjects assigned to DoD items and verified the correctness of assignment. During a meeting of the authors, no change was introduced, but it was decided that one item can tackle two subjects.
Step 4: Similarity and variability analysis—the first author extracted statements from same or similar DoD items and noted them down using the NoRT notation (Kopczyńska et al., 2018). Next, the second author went through the statements and DoD items and was to verify if he can formulate each DoD item using the assigned statement. During a meeting of the authors to discuss the results of their work no change was introduced.

4.9. Validity threats

The analysis of validity threats is based on the guidelines provided by Wohlin et al. (2012).

Construct validity. We identified several threats to construct validity of our study. The first one relates to the understanding of the term "Definition of Done" by the participants. To mitigate this threat, we provided a definition of DoD (based on the Scrum Guide) in the questionnaire. It does not guarantee that all participants had a common understanding of that term; however, it should prevent the situation when a person confuses the term with some other related concepts (e.g., acceptance criteria).

Since none of the previous studies reported comprehensive lists of benefits or problems related to using DoDs, there is a threat that our list is largely incomplete or partially irrelevant. To mitigate this threat, we based the proposed list of benefits and problems on the literature regarding DoD/requirements and brainstorming sessions among the authors of this paper. We also allowed for the "other" answer so that the respondents could name the benefits and problems that were not on our lists. Therefore, if our list was missing some important benefits or problems, we would most likely be informed about it by at least some participants.

As a respondent might not be aware of a certain benefit or problem, we allowed for the "I don't know" answer to the multichoice questions regarding benefits and problems. Also, we introduced a measure called *Awareness (A)* that is calculated as the percentage of the number of answers other than "I don't know" with respect to the total number of answers for each question regarding benefits or problems. Observing low *Awareness* would indicate that the respondents had difficulty in reliably assessing whether or not a given benefit/problem appeared in their project. This could be caused by a vague definition of the benefit/problem, respondents' lack of knowledge, or the elusive nature of the benefit/problem.

We asked the participants if the problems they encountered had *significant* negative impact on their projects, and if using DoD was *valuable* for their projects. We have to accept the fact that every person might have a different understanding of what these terms mean in this context.

Also, we made sure that the participants understood the goal of the study by clarifying the goal at the beginning of the questionnaire, so they were motivated to provide comprehensive and true answers to the questions.

Finally, we decided to run the survey anonymously, only after respondents completed all research-related questions they were asked to voluntarily provide their email address to receive a report of the results (still it gave the possibility to stay anonymous). In this way, we mitigated the evaluation apprehension threat so that the participants were guaranteed that they could be sincere about all their answers.

To mitigate the risk concerning *Content validity*, i.e., the questions we asked our respondents are not representative of what they aim to measure, we asked our experts in the pilot study if they see any necessary changes to introduce to achieve the goals of our study. We need also to accept that we could have added some more questions to the questionnaire. However, there is always a trade-off between the number of questions respondents would be willing to answer and the thoroughness of answering the research questions. In our opinion and taking into consideration the exploratory goal of our study, we selected the most important questions and left the space for future research to explore, e.g., the causes of the identified problems.

Internal validity. Although we did not seek to establish causal relationships, we believe that there are some threats that we can classify as belonging to internal validity.

First of all, we partially relied on inviting members of agile social networks (LinkedIn, Facebook, MeetUp) to participate in our survey. As a result, it limited our control over the response collection process. Consequently, we were not able to determine neither the response rate nor who received our invitation. Also, there is a question about the trustworthiness of the respondents. However, since a high percentage of respondents (45%) left their email addresses to be informed about the survey results, we expect that the topic was interesting to them, and they had no reason to intentionally provide false responses.

Also, informing the participants about the results of the study was the only incentive we offered for participating in the study. It could have a double-edged impact on the responses we collected. The use of monetary incentives could have increased the response rate, however, it could also harm the quality of the responses since some of the respondents might have been interested in completing the survey to be rewarded rather than motivated by the will of sharing their opinions with the community.

The other threat concerns the skills required to fill in the questionnaire. We assumed that the respondents would not have problems in responding to an on-line survey which is created using one of the most popular survey tool (Survey Monkey) and consisting of a commonly-used type of questions. Moreover, we assumed that they are fluent-enough in English to understand the questions. We conducted a pilot study to ensure that the questionnaire is easy to understand.

We continuously monitored the process of filling in the survey (using a quick analysis of the answers in the survey tool) and, especially, the time that the respondents spent on answering the questions. We did not observe any disturbing cases and it took ca. 17 min. 4 s (median) to complete the survey, which seems to be a reasonable duration for this kind of survey (we informed the participants on the first page of the questionnaire that the estimated time of completing the survey is between 15 and 20 min.). We also monitored those who dropped out. Since a significant proportion of respondents left the survey after the first question (see the analysis in Section 4.7) it seems that either the topic was not interesting or they did not have experience in it, or the list of benefits of using DoD discouraged the respondents from further work (e.g., seemed time-consuming or difficult). Thus, we might suspect that those who provided complete answers were those most interested in the topic.

External validity. The main threat to external validity concerns the representatives of the respondents and their projects. As it follows from our study design (the goal to draw an overall view of how DoDs are used) and the analysis of the demographic data (see Section 6) the respondents represent diverse profiles of software project participants (i.e., they have different experience, work in various industry sectors, projects were developed in different countries, etc.), which is essential to mitigate the risk of skewing the observations towards some particular context. The sample seems appropriate for the goal of our study, which was to get a general overview of the potential benefits and problems related to using DoDs in agile projects. However, a side effect of surveying such a broad population is that we were not able to relate certain benefits or problems to the presence of specific context factors in the projects. Therefore, based on our results, we cannot tell which benefits/problems one should expect to see in their particular agile project.

Also, we focused only on recent projects (i.e., the last time a respondent participated in the project was no earlier than 2015), which to some degree narrowed down the population under study and might introduce some selection bias to our study. However, our focus was not to study how the implementation of the DoD practice evolved over the years, but rather to study the current trends of how it is used and how it impacts agile software development projects.

Conclusion validity. The threat concerns the scales used to evaluate the benefits and problems of using DoD. They are subjective and could be interpreted differently by respondents depending on their knowledge, experience, character, etc. (this is also a threat to construct validity). We also allowed for providing open-text answers or stating "I don't know" to avoid biasing the results by forcing the respondents to answer about the provided sets. We employed a qualitative coding technique to analyze open-text responses. Although, three authors of the paper performed a multi-step process of analyzing the responses (see Section 4.8 for details), such an approach might have introduced some bias to the conclusions.

Table 3
Respondents' characteristics.

(a) Responsibilities of the respondents in projects

Responsibilities
(N = 137, multiple choices allowed)

Scrum Master's tasks	68	Designing software	31
Programming	60	Testing/QA	25
Project management	56	Other	22
Requirements	40		

(b) Experience of the respondents in projects

Experience
(N = 137, single choice allowed)

0–1 year	4	5–10 years	14
1–3 years	27	over 10 years	77
3–5 years	15		

5. Demographic information

As it can be seen in Table 3 (a), the survey respondents performed a large variety of project roles. However, most of them were responsible for Scrum Master-related and management-related tasks, programming, or requirements elicitation and analysis. Also, more than 66% of the participants had 5 or more years of experience, while only 3% of them worked in IT for less than a year.

The respondents mainly referred to their recent projects since 96% of them were developed within the last three years (69% in 2020, 24% in 2019, and 4% in 2018).

Fig. 3 presents the countries in which the respondents' projects were developed. The most dominating areas were North America, Europe, India, and Australia, while the two most underrepresented regions were China and Central Africa. A similar geographic distribution of responses was reported by other recent surveys regarding agile methods (Digital, 2021; Ochodek and Kopczyńska, 2018). However, we see that, in our case, Central Europe could be overrepresented in the Europe region.

As it is presented in Table 4, the projects were developed for different business domains, with banking and finance being the two dominating domains (Table 4a). In addition, the two most frequently developed types of applications were financial systems and web applications (Table 4b). However, we can see that the responses quite evenly covered most of the application types. The distribution of domains we observed in the studied sample is similar to the one reported by Hill (2011) for the ISBSG database.

By looking at the project teams, we can see that the vast majority of them worked in Scrum (89%), confirming that the DoD practice is strongly related to this agile framework (Table 4c). In other surveys regarding agile methods, Scrum was reported to be used by 67% (Digital, 2021; Matharu et al., 2015) to 94% (Ochodek and Kopczyńska, 2018) of the participants.

In nearly 60% of the projects, there were more than ten people involved (Table 4d). Therefore, we can suspect that many of the projects were developed in multiteam environments.

The analysis of the collected demographic information did not reveal strong evidence against the sample representativeness of the target population. However, we can see that our sample of projects developed in Europe could be slightly skewed towards Central Europe (see Fig. 3).

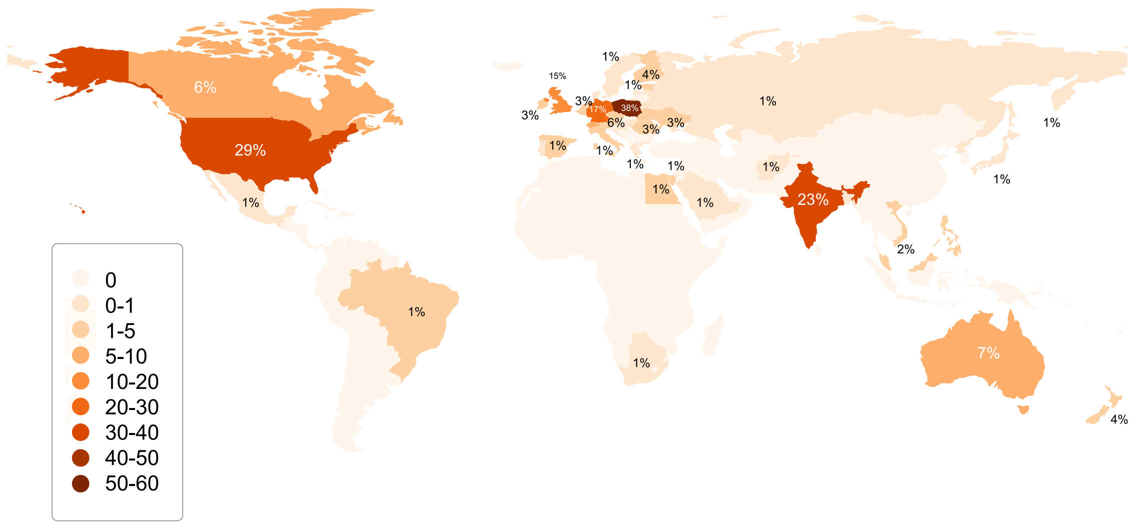


Fig. 3. The countries in which the respondents' projects were developed.

Table 4
Projects' characteristics.

(a) Domains in which the respondents' projects were conducted			
Domains (N = 137, multiple choice allowed)			
Banking	33	Government	5
Financial	16	Insurance	4
Services	13	Trading	3
Telecommunications	12	Construction	2
Medical & health care	12	Entertainment	1
Manufacturing	9	Others	20
Electronics & computers	7		
(b) Types of applications developed in the respondents' projects			
Types of applications (N = 137, multiple choice allowed)			
Financial	18	Sales and marketing	10
Web or e-business	18	Logistics	4
Document management	16	Trading	3
Management information (MIS)	13	Mobile application	1
Transaction or production	13	Personnel	0
Electronic data interchange	12	Other	0
Billing	11		
(c) Methodologies used in the respondents' projects			
Methodologies (N = 137, multiple choices allowed)			
Scrum	122	DSDM	1
Kanban	8	XP	2
Other	4	Crystal Clear Methods	0
(d) Sizes of teams in the respondents' projects			
People participating in projects (N = 137, single choice allowed)			
Up to 3 people	4	19–27 people	21
3–9 people	52	Over 27 people	27
10–18 people	33		

6. Results and discussion

6.1. Process of creating and maintaining DoDs

The results of the analysis of the open questions (regarding the process of creating and maintaining DoDs) and three closed questions (concerning the roles involved in that process, the frequency of updating DoDs, and the ways of publishing them) are summarized in Fig. 4. Also, the figure reports the number of answers to the multichoice questions and the frequency of codes resulted from the analyses of the responses to the open questions. However, since responding to the open questions was optional, and we cannot guarantee the completeness of each provided answer (i.e., mentioning all relevant concepts in the considered project), we will not attempt to evaluate the importance or popularity of the mentioned concepts based on the frequency of the codes. Still, the frequency analysis is possible for multichoice questions.

Based on the analysis of the codes regarding the process of creating DoDs, we identified that the activities mentioned by the participants create a process consisting of three stages (I–III) and five steps (1–5) (see Fig. 4 Part How): I. (1) proposing a DoD, (2) analyzing DoD items, (3) adjusting DoD items, II. (4) documenting, agreeing on, and publishing the DoD, and III. (5) evaluating and improving the DoD.

Forty-four respondents stated that the process of creating DoD in their projects started from preparing a DoD proposal, e.g., by a single team member "*Single member post item in backlog with DoD*", or by the whole team "*Initially look at what Utopia would look like*". Also, a few of them shared the triggers that initiated the process. The process of creating DoDs could be **triggered** by project events like project planning or code reviews, discussion between team members, or be a result of a single person's initiative (either a customer or team member). According to the respondents, the work on a DoD is equally distributed between the early stages of product development (e.g., project planning or kick-off meetings) and Sprints (e.g., Sprint planning, retrospective, and backlog refinement meetings). The most often used **toolbox (means)** of supporting this process are teamwork techniques like brainstorming (e.g., "*We sit together and defined it*", "*Brainstormed possible items*) or reusing knowledge from previous projects in the form of templates, organization standards, or DoD items from the past projects (e.g."*I copy-paste DoD from other projects as good starting point*"). Some DoD items are also defined based on requirements (e.g., based on non-functional requirements).

The second step of the DoD process (marked also with green in Fig. 4), according to 20 participants, is a further analysis after the initial proposal is created. The goal is to clarify ambiguities, assess the feasibility of the DoD items, prioritize them, and, finally, select those that should be preserved (e.g., "*then discussion with Team where still adjustments can happen*", "*then voted on must-haves vs. nice to haves*", "*then a dedicated "refinement session" on DoD is organized (the sooner the better) in which shortcomings/limitations/inflexibilities and other faults are discussed and*

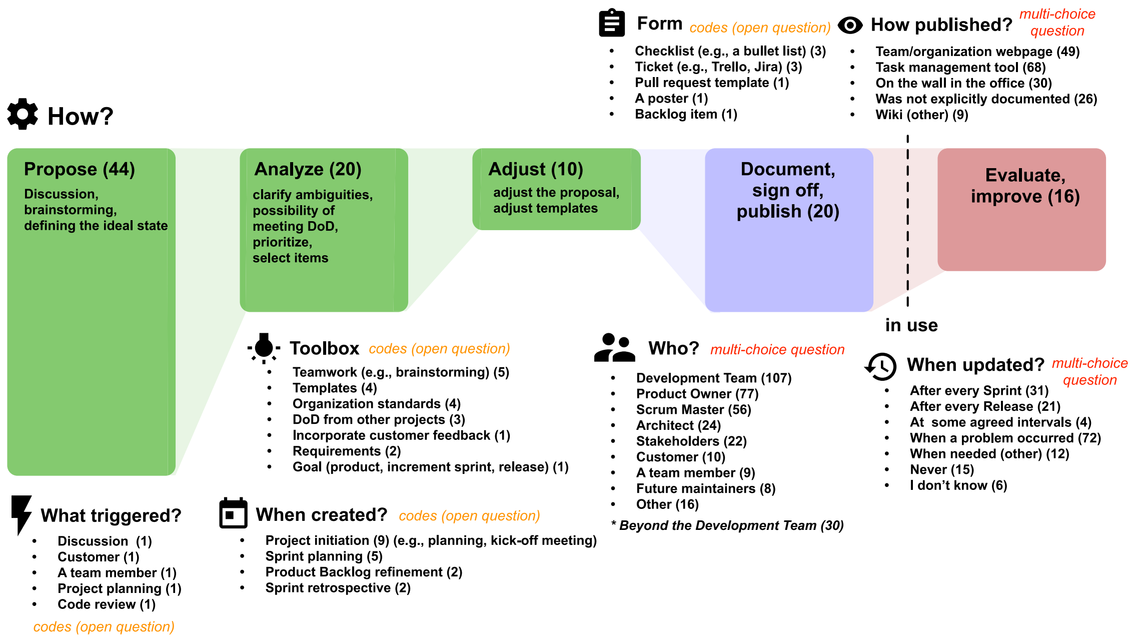

How?
Propose (44)
Discussion, brainstorming, defining the ideal state
Analyze (20)
clarify ambiguities, possibility of meeting DoD, prioritize, select items
Adjust (10)
adjust the proposal, adjust templates
Document, sign off, publish (20)
in use
Evaluate, improve (16)
Form codes (open question)
Checklist (e.g., a bullet list) (3)
Ticket (e.g., Trello, Jira) (3)
Pull request template (1)
A poster (1)
Backlog item (1)
How published? multi-choice question
Team/organization webpage (49)
Task management tool (68)
On the wall in the office (30)
Was not explicitly documented (26)
Wiki (other) (9)
Toolbox codes (open question)
Teamwork (e.g., brainstorming) (5)
Templates (4)
Organization standards (4)
DoD from other projects (3)
Incorporate customer feedback (1)
Requirements (2)
Goal (product, increment sprint, release) (1)
Who? multi-choice question
Development Team (107)
Product Owner (77)
Scrum Master (56)
Architect (24)
Stakeholders (22)
Customer (10)
A team member (9)
Future maintainers (8)
Other (16)
* Beyond the Development Team (30)
When updated? multi-choice question
After every Sprint (31)
After every Release (21)
At some agreed intervals (4)
When a problem occurred (72)
When needed (other) (12)
Never (15)
I don't know (6)
What triggered?
Discussion (1)
Customer (1)
A team member (1)
Project planning (1)
Code review (1)
codes (open question)
When created? codes (open question)
Project initiation (9) (e.g., planning, kick-off meeting)
Sprint planning (5)
Product Backlog refinement (2)
Sprint retrospective (2)

Fig. 4. Process of creating and maintaining DoDs (codes from the axial coding and answers to multi-choice questions). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

resolved"). After analyzing the proposed DoD, teams introduce the necessary changes. Once the DoD document is ready, it is made available in the second stage of the DoD process. According to the respondents, it usually has a **form** of a checklist that is **published** on the team's web page or wiki. Another form is to store it as "tickets" in task management tools or as backlog items. Some teams prefer to have DoDs printed out and hang on the wall. Also, some parts of DoD could be expressed in other forms—e.g., as pull-request templates. Still, 26 respondents stated that in their projects DoDs were not explicitly documented.

Finally, the last step of the DoD process (see Fig. 4 the last step colored with red), the respondents mentioned was about the need for further maintenance and improvement of DoDs in their projects. In most cases, the DoDs were updated on demand whenever there was a reason to do so (e.g., "*then retrospected at various points to make tweaks*"). However, performing regular DoDs reviews (e.g., after every Sprint or release) was also a frequently reported practice (e.g., "We refine/revisit occasionally in team retros"). Interestingly, 15 respondents stated that the DoDs in their projects have never been updated.

According to the respondents, the three roles defined in Scrum (Developers, Product Owner, and Scrum Master) are most often involved in the process of creating DoDs (multichoice question, see Fig. 4 Who?). However, we could still see that in 22% of the projects, the DoDs were created without involving the developers. It is a surprising observation since the DoD is used by developers everyday, and every increment they produce needs to adhere to it.

6.2. What DoD items are

24 respondents shared with us their DoDs, which in total had 143 DoD items. To better understand what DoD items are, we analyzed them on three levels: (1) categories of DoD items, (2) what DoD items are about, and (3) contents of DoD items.

First, we divided the DoD items into groups using the categories proposed by Silva et al. in Silva et al. (2017, 2018). We noticed that we had to extend the proposal by adding one more category, namely *AC check*, which would be used to group the items that concern the fulfillment of acceptance criteria.

As it follows from Fig. 5, the greatest number of DoD items concerned Quality Management (64) and appeared in almost all DODs (21 out of 24). The second most popular category was Process Management with 42 DoD items from 18 different DoDs. The category proposed by us, AC Check, was present in 7 DoDs in 15 DoD items. There were also classified over ten DoD items (11 to be precise) from 8 different DoDs to the Deploy category. However, only one DoD contained DoD items (3) concerning Regulatory Compliance.

Second, we decided to understand better what the DoD items are and extracted from each item its *subject* that is the thing about which is the DoD item. It follows from Table 5 that DoD items are most frequently about tests (29 items), code review (15), acceptance criteria (15), and documentation (12).

Third, we decided to better understand the contents of DoD items. We noticed that different teams (respondents) formulate either similar or the same DoD items. Thus, we focused on the similarities and variability in DoD items. To document the phenomenon we decided to use the NoRT notation, which allows to specify such differences in natural language (Kopczyńska et al., 2018). Each statement in this notation consists of: (1) the part that is the unchangeable core; (2) parameters represented by items in angle brackets (e.g., <number>); (3) alternatives (options) that can be selected, are presented in brackets and are separated with bar characters (e.g., (milliseconds | seconds | minutes)). Each option can be either a parameter, a static (a statement that remains unchanged), or a combination of the two. To allow choosing more than one option, the '}' option modifiers are used, and to allow omitting a certain option the option modifier ']' is used.

Table 6 contains 10 statements in the NoRT notation showing the similarities and variability of the DoD items. The statements are based on the DoD items most frequently mentioned by our respondents, i.e., each one is based on at least 3 DoD items (so it might be referred to as the top 10 most frequent). We added also examples of DoD items that constituted the basis for the statements, some of which were changed to make the contents anonymous.

All statements but one show both similarities and variability. The highest number of similar DoD items (13) concerned the code

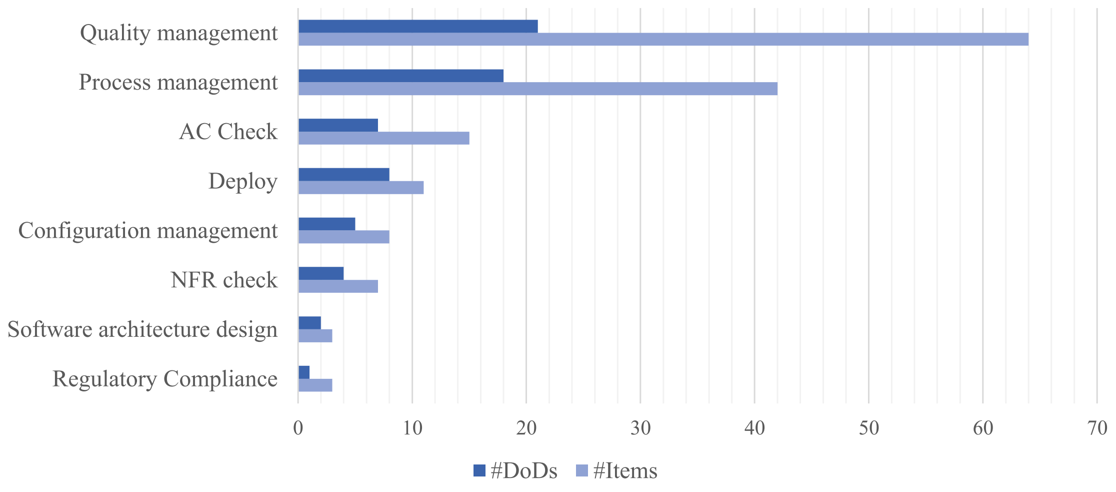


Fig. 5. Number of DoD items and DoDs per category of DoD item.

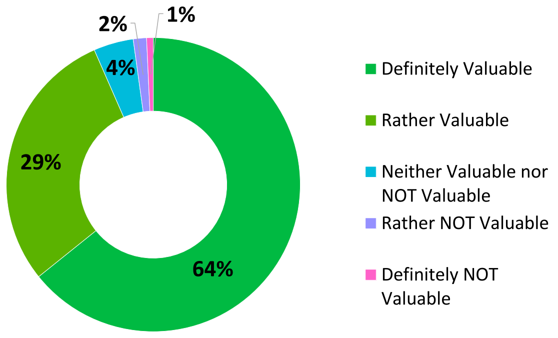


Fig. 6. Perceived value of using DoD.

review process. The S11 statement shows that teams either focus on a specific unit of work (e.g., task) or just generally state that the code must be reviewed. Some respondents added information about the positive result of the review process or about who will conduct the review. The S3 statement shows that different teams use different types of testing and sometimes focus their testing effort on a defined scope (like feature) or on a certain branch. No variability was identified in the DoD items about completion of coding — the S23 statement was formulated based on three same DoD items from different respondents.

6.3. Benefits of using DoDs

As it follows from Fig. 6, the great majority 93% (128) of the respondents regarded using DoD as valuable for their project ("Rather valuable" or "Definitely valuable"), while 64% of them (88) perceived it as definitely valuable.

To investigate how often certain benefits of using DoDs appear in the projects, we defined a measure called *Perceived Value (Val)*. It is calculated as the percentage of the answers confirming the presence of a given benefit in a project ("Definitely YES" or "Rather YES") with respect to the total number of the responses, but excluding the "I don't know" answers. Excluding the "I don't know" answers from the denominator means that we focused only on the cases for which the respondents felt competent enough to judge the causality between using the DoD practice and the presence of a given benefit. Please note that there was also a neutral answer available for the respondents ("Neither significant nor NOT significant"). Still, it does not mean that we completely ignored the "I don't know" answers. These answers were incorporated into the *Awareness* measure which is the percentage of the number of answers other than "I don't know" with respect to the total number of answers for each question. It complements *Perceived Value* by showing the level of support for the given benefit in the collected data. For instance, a benefit with very high *Perceived Value* and very low *Awareness* should be taken with a grain of salt (especially, since we search for benefits that are common across different project contexts). Of course, there might still be several valid reasons for making such an observation (e.g., a benefit might be visible only to some specific roles, or it could be intangible by its nature—be something difficult to measure).

The top five benefits of the highest Perceived Value concern organizational (management) and technical areas are (see Table 7): making work items complete, assuring product quality, ensuring that the activities other than coding were executed, ensuring that all quality gates are passed, and keeping the product releasable. All but one of the benefits appeared in over 50% of the projects. The one exceptional case was the B19 "Less time spent on manual testing" benefit. It appeared in around 1/3 of the projects (32%). Finally, the respondents extended the provided list of benefits by adding two new proposals: B20 "DoD helped with creating clean code" and B21 "DoD caused that the whole team felt the ownership of quality".

The calculated *Awareness (A)* (see the preceding paragraphs and Section 4.9 for the explanation of the measure) for the benefits ranged between 90% and 99%. Thus, we conclude that *the respondents were well-informed and knowledgeable about the consequences of using DoD*. The benefit with the lowest *Awareness* was: B9 "Increase in customer's satisfaction". The second lowest *Awareness* was observed for B17 "Balance between short-term delivery of features and long-term product quality". Both of these benefits might be difficult to judge by a project team member who is not closely collaborating with the customer or is not involved in the planning and assessment of product quality.

As we can see in Fig. 7, the benefits of using DoD seem to be highly correlated with each other (only positive correlations were observed). The calculated Spearman's ρ for the statistically significant relationships ranged between 0.13 and 0.75 (mean = 0.39). According to Akoglu (2018) the mean value of $\rho = 0.39$ could be interpreted as a moderate/strong relationship, while the strongest observed relationships could be classified as moderate to very strong.

The strongest correlation (ρ between 0.64 and 0.75) was observed between a group of three benefits related to code quality (B11: ensuring coding standards, B16: transparency of code quality, and B18: clean repositories). The second highly correlated

Table 5
Categories and subjects of DoD items with the number of items and number of DoDs in which they were present (multiple labels allowed).

Subject of item	# DoD items	# DoDs
Quality management		
Tests	29	16
Code review	15	13
Code	4	4
Feature	4	3
Defect	4	3
(Key) Metrics	2	2
Test coverage	2	2
Increment	1	1
Monitoring	1	1
Review	1	1
Technical debt	1	1
Process management		
Documentation	12	8
Signoff	7	5
Feature	4	4
Tasks	4	4
Demo	4	3
Code	3	3
Release	2	2
User stories	2	2
Knowledge of team	2	1
Assessment	1	1
Briefing	1	1
AC check		
AC	15	7
Deploy		
Deployment	7	4
Release	3	3
Release notes	1	1
NFR check		
NFR	6	3
NFRs	1	1
Configuration management		
Merge	3	3
Build	2	2
Code	1	1
Feature	1	1
Repository	1	1
Regulatory Compliance		
External standard	3	1
Software architecture design		
Design	2	1
API documentation	1	1

group of benefits (ρ ca. 0.61) regarded the quality-assurance process (B2: helping assure the quality of product, B3: ensuring that the activities other than coding are executed, and B4: ensuring that all quality gates are passed). Finally, one more correlation with $\rho \geq 0.60$ was observed that shows a relationship between the benefit of a smaller number of defects being released (B8) and the increase in customer satisfaction (B9). Finally, there were only two benefits that did not correlate visibly with many other benefits, i.e., less time spent on manual testing (B19) and making team members aware of the current project status (B10). Unfortunately, since so many benefits were highly correlated, the number of identified association rules did not allow us to reveal any additional interesting and meaningful groups of benefits.

6.4. Problems resulting from not using DoDs

When deciding on adopting a practice, it is important to know what benefits it might bring to the project, however, it is also useful to understand the consequences of neglecting it.

Thirty-one respondents shared some problems they had encountered in their other projects that occurred due to the lack of DoD. Those aspects concern technical-, team-, and project-level issues and are listed in Fig. 8. The top five most frequently mentioned problems were:

- not meeting deadlines (e.g., "*we went beyond the timeline and budget*"),
- lack of shared understanding of what done means (e.g., "*Some member have different view of what to achieve, which sometimes resulting in over commit to task, team member fighting on what task to cover in each feature*", "*Not having alignment on what "done" means for a team causes lots of problems and friction between dev team and PO and team and stakeholders.*"),
- defects, low quality, and technical debt (e.g., "*technical debt creeping in*", "*they haven't delivered the expected quality and ended up in lot of rework post UAT*", "*Finally, defects were commonplace*").

6.5. Problems encountered while using DoDs

We analyzed the responses regarding the problems that might appear while using DoDs by investigating how often certain issues appeared in the projects and how harmful they were. In particular, we calculated two measures that allowed us to quantify these two aspects:

- *Frequency (Freq)* is the percentage of all responses except for the "Problem did not appear" and "I don't know" with respect to the total number of responses other than the "I don't know" answers;
- *Significant impact (Sig)* is the percentage of the responses claiming that the problem had a "Definitely Significant" or "Rather Significant" negative impact on the project with respect to the total number of responses other than the "I don't know" answers.

Freq tells us how likely it is that a given problem appears in a project which uses the DoD practice. We excluded the "I don't know" answers for the same reasons as we did it when calculating *Perceived Value* (see Section 6.3). These answers can either mean that the respondents had not encountered a given problem before or that they had insufficient knowledge to confirm its presence or lack of thereof. *Sig* allows us to evaluate the severity of problems. We focused only on the situations when a given problem leads to serious negative consequences. However, even when the respondents stated that the impact of a given problem was not significant, it does not mean that it was negligible.

As it follows from Table 8, the top 5 most frequent problems (P1 "It was difficult or time-consuming to implement DoD item(s)", P2 "Some team members had different understanding of DoD", P3 "It was difficult or time-consuming to verify if DoD item(s) are satisfied", P4 "Problem with defining universal DoD", and P5 "Some DoD item(s) were imprecise") were reported to appear in 70% or more of the projects. The next two frequently appearing problems were missing (P6), unclear/difficult to understand (P7), or impossible to verify (P8) DoDs, which were reported to be present in over 60% of projects. The least frequent problems, i.e., infeasible DoD items (P19), incorrect DoD items (P18), and not written down DoD (P17) were present in more than 45% of the projects.

With respect to significance, we observed that the problems concerning the unavailability of DoD (P17), imprecise DoD items (P5), missing DoD items (P6), and reaching an agreement between the team and stakeholders (P12) were perceived as having a significant, negative impact on 44%–48% of the projects.

Table 6
Statements showing variability and similarity in DoD items.

- **S11** (13 DoD items): `{Task | Code | <unit of work>} must be reviewed [and passed] [by {at least one other person, two reviewers}]`
  e.g., *Code must be reviewed by at least one other person*
- **S3** (11): `{Unit | Integration | Regression | Manually | System | <type of test>} tested [{feature | <scope of test>}] [on branch <name>]`
  e.g., *Manually tested on branch develop*
- **S14** (9): `[{Functional | Technical | Integration | Deployment instructions | <name of documentation>}] Docu- mentation is completed.`
  e.g., *Technical documentation is completed*
- **S19** (9): `The acceptance criterion is satisfied: <contents of the AC>`
  e.g., *The acceptance criterion is satisfied: sorting table by all columns*
- **S2** (7): `{Functional | <type of test>} Tests were executed [and pass] [in <environment name>]`
  e.g., *Functional tests were executed and pass*
- **S22** (5): `Deployed to <environment name>`
  e.g., *Deployed to development environment*
- **S24** (4): `No {open defects | <type of issue>} present`
  e.g., *No open defects present*
- **S25** (3): `Released [to <environment name>]`
  e.g., *Released to production*
- **S23** (3): `Coding is complete`
- **S17** (3): `<unit of work> must satisfy the acceptance criteria`
  e.g., *Feature must satisfy the acceptance criteria*

Table 7
Benefits of using Definition of Done.

ID	Benefit from using DoD	Val%	A%
B1	Help to make work items complete	93	99
B2	Help to assure quality of product	92	98
B3	Help to ensure that the activities other than coding were executed (e.g., code review, manual testing, build)	90	99
B4	Help to ensure that all quality gates are passed (e.g., performance testing, code review, security check)	90	98
B5	Help to keep product releasable	86	97
B6	Help in effort estimation	79	98
B7	Promotion on the meaning of "complete" work between stakeholders	79	96
B8	Fewer bugs and issues get released	76	98
B9	Increase in customer's satisfaction	72	90
B10	Help to make team members aware of the current status of the project	66	97
B11	Help to ensure organization's coding standards	66	97
B12	Reduction of technical debt (early spotting the defects)	64	98
B13	Increase of Development Team productivity	63	96
B14	Reduction in time for reworks of implemented features	61	96
B15	Help to keep the documentation up-to-date	56	96
B16	Transparency of code quality (e.g., explained what it is)	55	96
B17	Balance between short-term delivery of features and long-term product quality	55	93
B18	Help to keep code repository clean	52	98
B19	Less time spent on manual testing	32	96
[a]B20	"*DoD helped with creating clean code*"	–	–
[a]B21	"*DoD caused that the whole team felt the ownership of quality*"	–	–

[a]Benefits identified by some the participants (using the "other" option).

Table 8
Problems with using Definition of Done.

ID	Problem with using DoD	Freq%	Sig%	A%
P1	It was difficult or time-consuming to implement DoD item(s)	77	33	97
P2	Some team members had different understanding of DoD	76	40	100
P3	It was difficult or time-consuming to verify if DoD item(s) are satisfied	75	27	97
P4	Problem with defining universal DoD (e.g. DoD on organization level)	71	40	94
P5	Some DoD item(s) were imprecise (loosely stated goals)	70	47	97
P6	Some DoD item(s) were missing	66	44	99
P7	DoD item(s) were unclear or difficult to understand	65	38	97
P8	Some DoD item(s) were impossible to be verified	65	30	97
P9	Some activities were omitted as DoD stated that they do not have be executed	62	37	97
P10	Too extensive set of DoD items	61	43	99
P11	Team members did not care about DoD (they omitted some or all DoD items)	59	43	99
P12	Problem with agreement between the Team and stakeholders on the DoD	55	45	98
P13	DoD was changing during Sprint/increment	55	39	99
P14	Some DoD item(s) were out of date	55	44	98
P15	Some DoD item(s) were irrelevant	53	39	97
P16	DoD was creeping (continuously growing in uncontrolled manner)	52	41	97
P17	DoD was not documented, unavailable (DoD was established verbally, not written down)	47	48	99
P18	Some DoD item(s) were incorrect	46	40	98
P19	Some DoD item(s) were infeasible (impossible to satisfied)	46	38	98

To evaluate the importance of the problems, we should consider both *Freq* and *Sig* measures. Fig. 9 presents each of the problems in the two-dimensional space of these two measures with arbitrary guiding lines added for each of the measures at 33%, 50%, and 67%. Based on the plot, we could state that the most important problems are P2, P4, P5, and P6 since they are

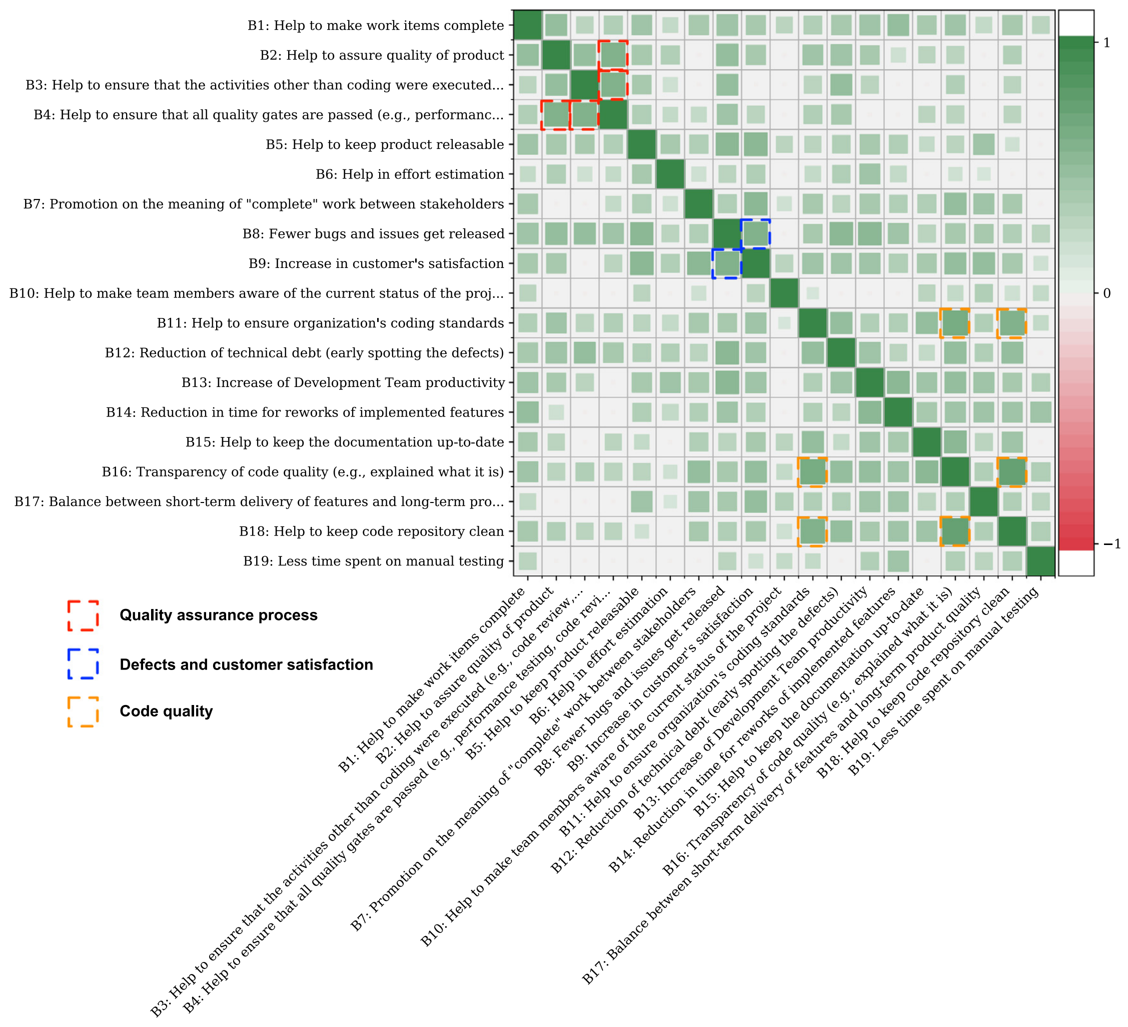
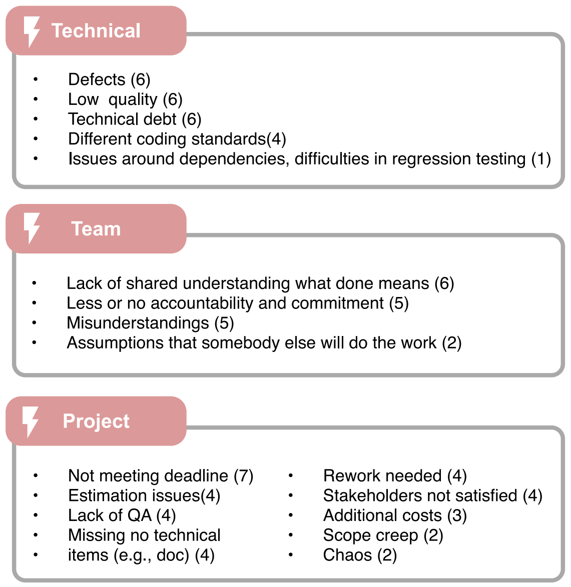


Fig. 7. Correlations between the reported benefits of using DoD (only statistically significant correlations are presented).


Technical

- Defects (6)
- Low quality (6)
- Technical debt (6)
- Different coding standards(4)
- Issues around dependencies, difficulties in regression testing (1)

Team

- Lack of shared understanding what done means (6)
- Less or no accountability and commitment (5)
- Misunderstandings (5)
- Assumptions that somebody else will do the work (2)

Project

- Not meeting deadline (7)
- Estimation issues(4)
- Lack of QA (4)
- Missing no technical items (e.g., doc) (4)
- Rework needed (4)
- Stakeholders not satisfied (4)
- Additional costs (3)
- Scope creep (2)
- Chaos (2)

Fig. 8. Problems that software projects struggle with when the DoD practice is not used.

all frequently appearing and have significant impact. All of them relate to the issues with specifying DoDs (missing, imprecise, or ambiguous items that are understood differently by team members).

To investigate if the findings concerning the problems are reliable, similarly like in the case of the earlier analysis of benefits, we calculated the *Awareness (A)*, which was between 94% and 100%. Thus, it seems that the respondents were also well-informed and knowledgeable to answer the questions about the problems concerning using DoD.

Fig. 10 shows the correlation matrix for problems while using DoD. The calculated Spearman's ρ for the statistically significant relationships ranged between 0.17 and 0.80 (mean = 0.39). The mean value of $\rho = 0.39$ could be interpreted as moderate/strong relationship while the strongest observed relationships could be classified as strong to very strong. The analysis of the frequent itemsets, association rules, and the calculated correlation coefficients allowed us to identify a group of eight problems with strong intra-group correlations. All these problems relate to how DoD-items are formulated/specified: imprecise (P5), missing (P6), unclear (P7), impossible to verify (P8), out of date (P14), irrelevant (P15), incorrect (P18), and infeasible (P19). When we combined all these problems into a single meta-problem, it appeared to be correlated with many other problems (at least one of the problems covered by the meta-problem was correlated with

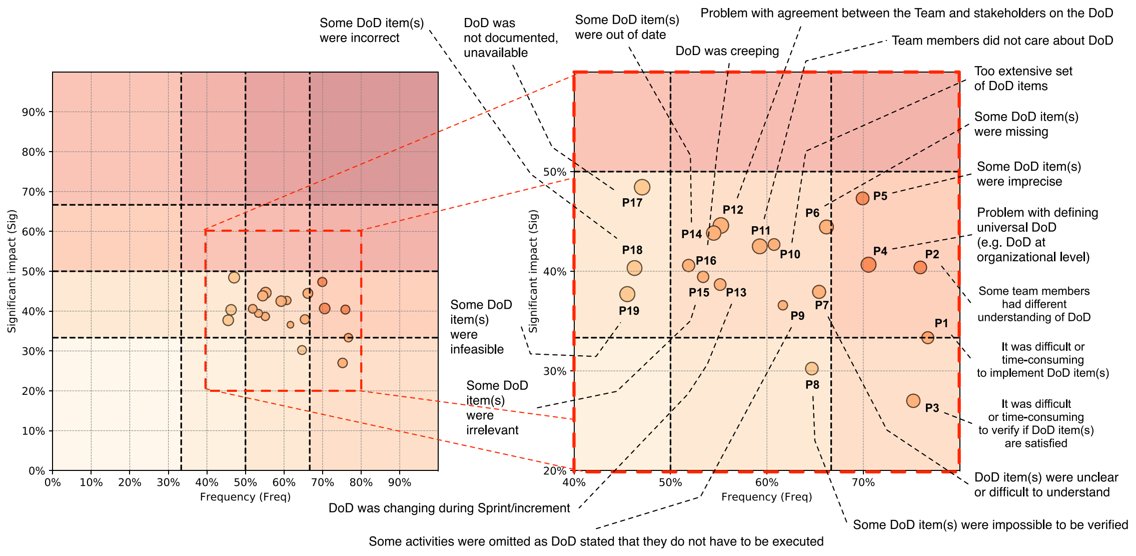


Fig. 9. The importance of identified problems related to using DoDs (color—importance, size—the number of responses assessing the negative impact of problems as definitely significant). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

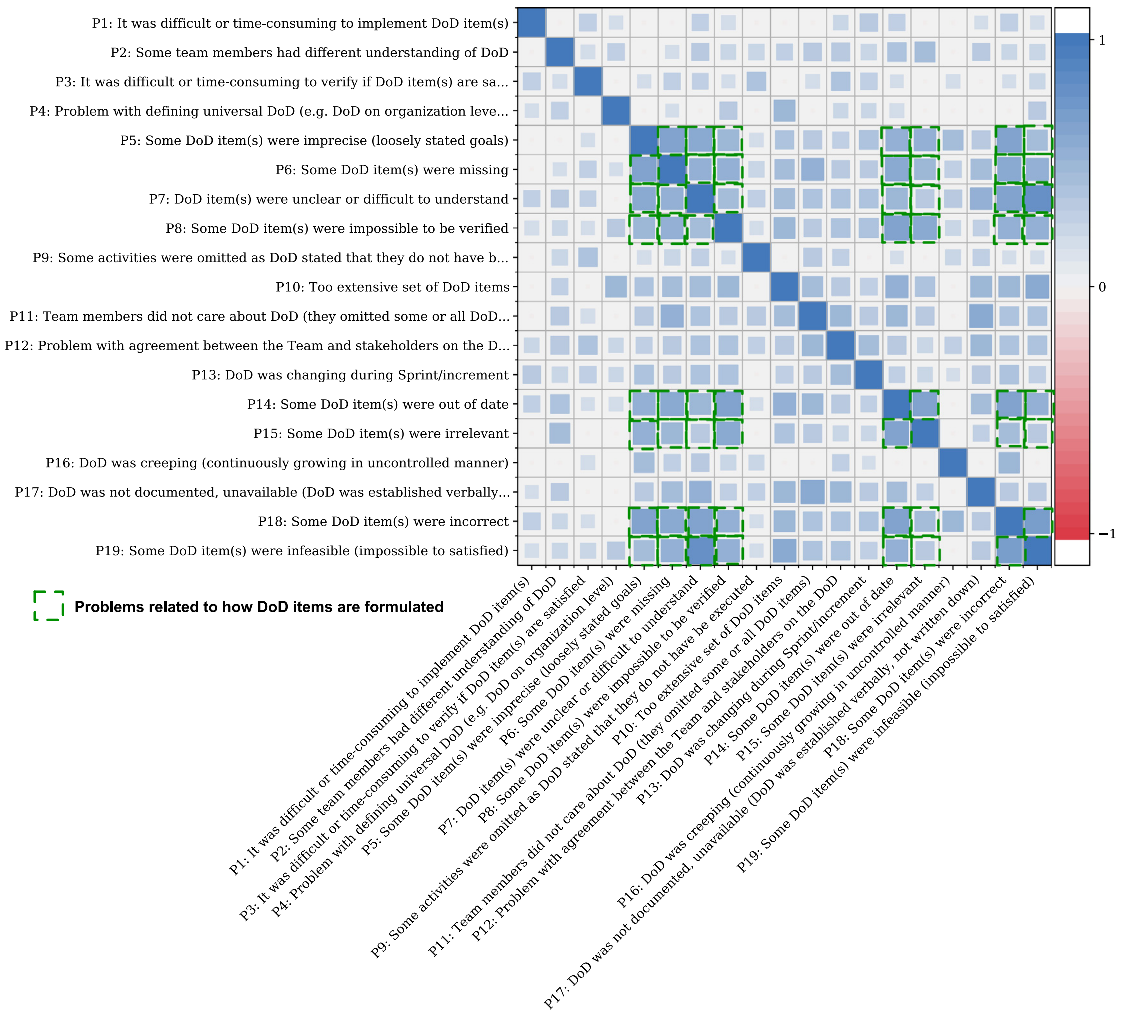


Fig. 10. Correlations between the reported problems while using DoD (only statistically significant correlations are presented).

another problem), e.g., a different understanding of DoD (P2), difficulties and effort while verifying DoD satisfaction (P3), too extensive DoD (P10), or problems with the agreement between team and stakeholders on DoD. We were also able to identify some other relationships by analyzing Fig. 10 and association rules. For instance, in the association rules having "P2: Some team members had a different understanding of DoD" as consequent, the antecedents were (apart of the previously mentioned meta-problem with how DoD items are formulated) problems like a too extensive set of DoD items (P10), problems with agreeing on DoD within stakeholders and team (P12), difficulties and effort required to verify DoD satisfaction (P3), or problems with defining DoD at the organizational level (P4). Although these relationships are derived based on correlations, we could hypothesize that some causality relationships between these problems exist. For example, if a set of DoD items is too extensive, it might be hard to memorize it, and consequently, it could lead to a different understanding of the DoD by team members. Another example could be "P10: Too extensive set of DoD items" which was a consequence of rules having antecedents such as problems with defining a DoD at the level of organization (P4), omitting some activities defined in DoD (P9), problems with the common understanding of DoD (P2), and the meta-problem regarding how DoD is formulated. One more example would be "P12: Problem with agreement between the Team and stakeholders on the DoD" which was a consequent of rules having antecedents such as difficulties in verifying DoD items satisfaction (P3), a different understanding of DoD by team members (P2), problems with creating universal DoD (P4), changing DoD during iterations (P13), or when team members did not care about DoD (P11).

Fortunately, many of the problems could be resolved. Twenty-three respondents shared their experience on how the problems concerning DoD were solved in their projects. The majority of them (11) explicitly stated that the problems were solved by team (e.g., "*Fix them! It's just a matter of sitting down with the team*", "*we called in a team meeting and worked out issues*". Two respondents reckoned that a person with a coach/Scrum master role was needed to support the team or initiate the process. Six respondents stated that the problems were solved while team members were interacting, mostly while team meetings/workshops such as retrospectives. What is interesting is that two respondents highlighted that solving problems with DoDs is not a "one-shot" activity, but an incremental process since DoDs evolve in time. During the process of solving the problems, DoD items were clarified (5), "*started to have shared understanding*" (2), made relevant (2), adjusted (1), discussed (1), negotiated (1), made useful (1), some external dependencies were removed (1), or at one extreme the DoD was removed (1).

6.6. Problems and benefits relationships

The survey questions asked directly about the problems that appeared in the participants' project while using DoD and about their significance. However, by analyzing the relationships between the problems and benefits, we can investigate whether the benefits from using DoD are less/more likely to be observed when certain problems with DoD materialize (or alternatively, whether they are independent).

The correlation matrix in Fig. 11 shows that both positive and negative (statistically significant) correlations could be observed between problems and benefits. The calculated Spearman's ρ ranged between 0.18 and 0.32 (mean = 0.23) for positive correlations and between −0.17 and −0.33 (mean = −0.23) for negative ones and could be interpreted as weak to moderate relationships.

Observing negative correlations means that the more significant problems with using DoD are observed, the fewer benefits one should see in a project. Most of the benefits are correlated with at most a few problems. Benefit "B4: Help to ensure that all quality gates are passed" is an exception since it is negatively correlated with nearly all the problems. Another benefit that is negatively correlated with numerous problems (seven) is "B2: Help to assure quality of product". Both of these benefits are related to the quality assurance process and both correlate with the problem "P11: Team members did not care about DoD (they omitted some or all DoD items)". We hypothesize that the fact that a team does not care about DoD might be an indicator that the team could be also reluctant to perform other pro-quality activities. Both of these benefits are also correlated with some problems related to how DoD is formulated (our meta-problem). Finally, we could see that benefit "B8: Fewer bugs and issues get released" is negatively correlated with problems such as "P7: DoD item(s) were unclear or difficult to understand", "P8: Some DoD item(s) were impossible to be verified", "P17: DoD was not documented, unavailable (DoD was established verbally, not written down)", and again with "P11: Team members did not care about DoD (they omitted some or all DoD items)". We hypothesize that there could be some causality between the problems and benefit B8 since items that are unclear, difficult to understand (P7), impossible to verify (B8), and agreed only verbally (P17) could make the testing process difficult (not to mention that the presence of problem P11 characterizes the team's attitude to quality assurance).

The presence of positive correlations between problems and benefits might seem surprising at first, however, we believe that some of these relationships could be justified. For instance, observing positive correlations between "P1: It was difficult or time-consuming to implement DoD item(s)" and numerous benefits show that "quality does not come for free" and teams need to invest time and resources into pro-quality activities to see the benefits. Another reason for observing positive correlations could be that the appearance of some DoD-related problems might be a good indicator of deeper problems in projects or organizations (e.g., problems with communication, decision making, or resigning from pro-quality activities)—DoD could be perceived as even more valuable by teams that struggle with quality assurance. Finally, the problems with using DoD could be temporal, and in some cases, their presence could even trigger positive changes and lead to implementing corrective actions.

7. Study implications

7.1. Implications to research

Our study confirms that using DoDs is useful, as has already been claimed by several researchers and practitioners based on their experience, e.g., Saddington (2012), Jakobsen and Johnson (2008) and Cohn (2010).

We identified and evaluated the importance of several benefits that DoDs might bring. However, our findings are a trigger for opening a new discussion about when such benefits are present and how to create an effective DoD to help materialize these benefits.

Our study extends the body of knowledge related to the problems encountered while using DoDs by providing an initial evaluation of their frequency and significance. Since several problems might appear frequently and have a negative impact on agile projects, a further and more in-depth analysis would be valuable. We identify an opportunity for more fine-grained research on why the problems appear, what we can do to prevent them, and how to solve them.

Also, the results of the survey complement the previous studies on investigating how the practice of DoD is applied and what DoD items are, e.g., Silva et al. (2017, 2018). The respondents reported that there are multiple ways of creating and maintaining DoDs. Therefore, as future research, it would be worth investigating the advantages and disadvantages of different approaches. Our analysis of 24 DoDs provided by the respondents showed that DoD items concern different subjects (like code, code reviews,

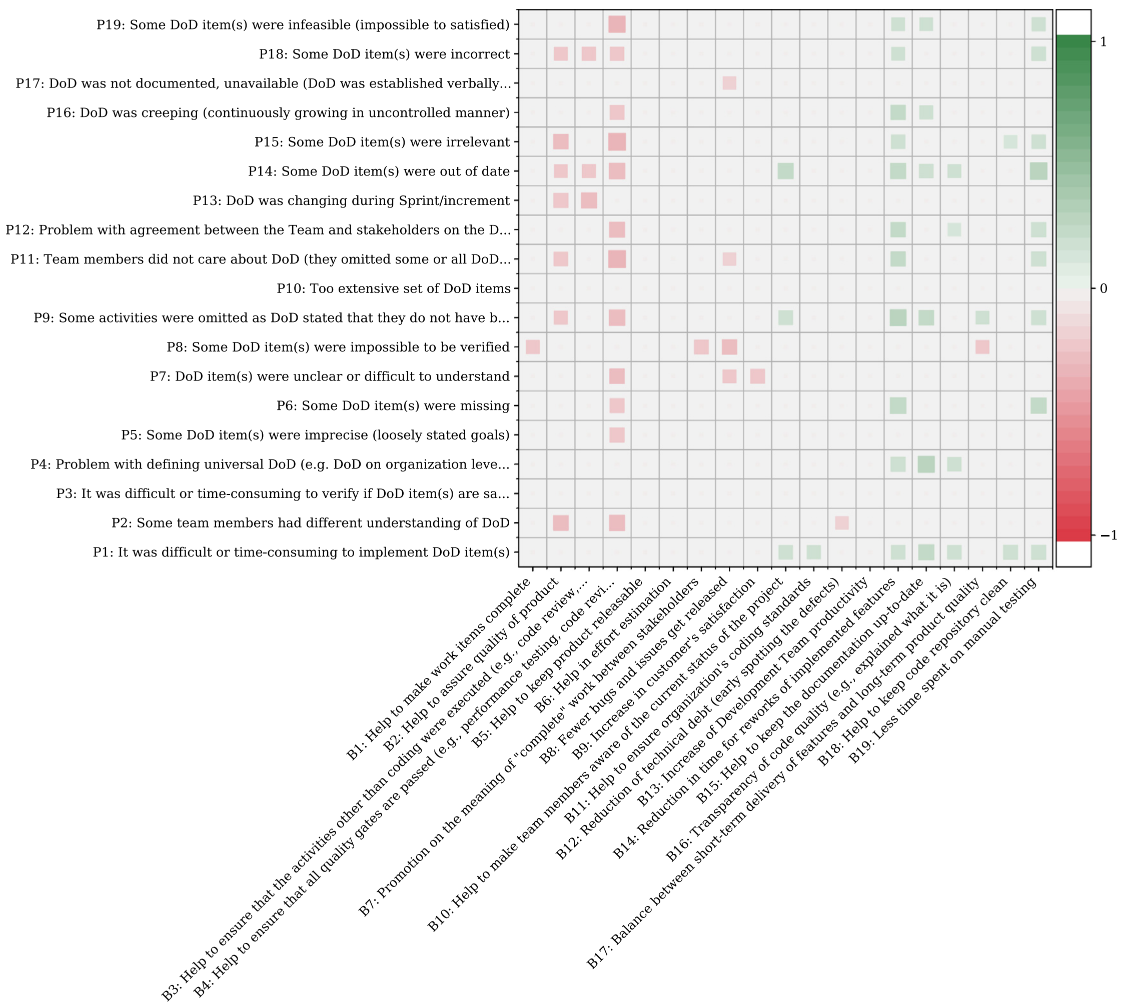


Fig. 11. Correlations between the reported problems and benefits of using DoD (only statistically significant correlations are presented).

tasks etc.) and their contents might be similar across different organizations. Thus, studying more DoDs might be valuable to broaden the knowledge about what DoD items are and how they are formulated. Possibly, it might additionally lead to learn how to efficiently and effectively specify DoD items.

Moreover, new research directions are opened by two groups of problems—the problems related to the specification of DoDs and the cost of creating and maintaining DoD. It would be worth considering the possibility of supporting or even automating to some degree the process of creating DoDs.

Finally, from the empirical software engineering perspective, we would like to draw the attention of other researchers to the usability of the questionnaire used in their survey, especially the presentation of the questions. It might have a significant impact on the response rate. It follows from the analysis of the drop-outs in our study that a significant number of our respondents left when they were presented a list of benefits or a list of problems to assess (i.e., a list of several items to assess on a scale with more than 5 values). It seems that it might be caused by the presentation method of the list-type of questions, e.g., the impression that answering the question would be overwhelming or time-consuming.

7.2. Implications for practitioners

Our findings show that the DoD practice is perceived as useful by nearly all practitioners who used it. The primary benefits are about helping to make work items complete to ensure product quality and that the activities other than coding are executed, and all quality gates are passed to keep the product releasable.

However, when deciding to use DoD or when thinking about improving that practice, one needs to understand that there might appear some problems along the way. The list of problems and their evaluation from the frequency and significance perspectives might help practitioners to assess the risks related to adopting the DoD practice and prepare in advance.

The results of our study showed that problems related to how DoD items are formulated correlate with nearly all other DoD-related problems (including those that could lower the chances of materializing benefits from using DoD). We believe that such problems can be easily mitigated by reviewing DoD documents. We propose a set of DoD quality criteria called VIRUPCUS that correspond to each of the aforementioned problems (**V**erifiable (P8), **I**dentified (P6), **R**elevent (P15), **U**p-to-date (P14), **P**recise (P5), **C**orrect (P18), **U**nambigious (P7), **S**atisfiable (P19)). The criteria can be used as a checklist. For instance, when reviewing one of the DoD items provided by respondents stating that *"burndown [should be] as close to deadline as possible"*, we could argue that it is *imprecise* and impossible to *verify*. Another example could be a DoD item stating that *"automated test [are] written"*. We perceive this DoD item as *incorrect* because we suspect that the real intent behind formulating it was to have automated test cases that pass. Also, it is *imprecise* because we do not know "how many" tests should be written. Finally, the word "written" seems *ambiguous* in this context—does it mean implemented?

It is also important to be aware of the fact that when a team dismisses to use DoD in their project, several important risks might materialize, such as, not meeting the deadline, lack of shared understanding of what "done" means, defects, low quality, and technical debt.

Moreover, the results of our study show how practitioners establish and maintain DoD which can be used for educational purposes. It might be especially useful for novice Scrum adepts who look for guidelines on how to create a DoD or for those who seek some ideas on how to improve their process. Together with the information about the benefits, it might provide a strong argument in favor of using DoD for those who are reluctant (e.g., for Scrum Masters or agile coaches who would like to convince those undecided).

Finally, another educational application of the results of our study could be for those looking for what their DoD might contain or what they can add to their DoD. Those practitioners can look into the categories or the subjects of DoD items to support their, e.g., brainstorming session in their team. Additionally, the ten statements that can be used to formulate DoD items might also trigger a discussion of whether they are relevant in their context or not.

8. Conclusions

The goal of the study presented in this paper was to investigate the usefulness of using the Definition of Done (DoD) practice. To achieve that goal, we conducted a questionnaire-based survey. In the research study 137 practitioners from all over the world took part.

Within the study, we looked at the usefulness of DoD from two perspectives: the benefits it might bring to a project or product team, and the problems it might cause. Additionally, we investigated how DoDs are created and maintained to extend the existing body of knowledge.

Our findings show that the vast majority of practitioners (93%) perceive DoD as at least valuable and confirm that its presence in the project help to make work items complete, assure product quality, ensure the needed activities are executed, ensure that all quality gates are passed, and help to keep product releasable. The usefulness of DoD can be also confirmed by an analysis of problems that might appear when DoD is not used. Our respondents mentioned defects, low quality, technical debt, lack of shared understanding of what done means, and not meeting the deadline as those most frequent.

On the other hand, the results of the survey show that there are some common problems related to using DoDs, and some of them might even significantly impact an agile project (e.g. high effort of implementing and using DoDs or different understanding of DoD items).

Moreover, the most important problems with using DoDs relate to imprecise, missing, or unclear DoD items. These problems are getting solved by agile practitioners mainly internally within a team through some collaborative activities like workshops or, in some cases, with the help of an agile coach. We proposed a set of DoD quality criteria (VIRUPCUS) that might be used as a tool to support reviewing DoD documents.

Furthermore, the respondents characterized the process of creating and using a DoD and what DoD items are. Although there are some variations in that process, the most general steps are: (1) propose a DoD, (2) analyze it, (3) adjust the proposal, (4) document, sign off and publish the DoD, (5) use it, (6) evaluate, and improve the DoD. In most cases, the DoD was created by the members of Scrum Team, however, we observed that in nearly 22% of the studied projects, the developers were not involved in that process. We also found that most DoD items concern quality management and process management, and are about tests, code reviews, documentation and acceptance criteria. The subjects and contents of some DoD items across different companies is the same or similar with some variability.

The findings from the study allowed us to identify new research directions, i.e., the necessity of in-depth analysis of the DoD practice, of the identification of the means of preventing or mitigating the problems related to using DoDs, and, finally, of the investigation of the ways to maximize the benefits that come from using DoDs.

CRediT authorship contribution statement

Sylwia Kopczyńska: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – Original Draft, Writing – review & editing, Visualization, Supervision, Project administration. **Mirosław Ochodek:** Conceptualization, Methodology, Software, Formal analysis, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Jakub Piechowiak:** Methodology, Formal analysis, Investigation, Resources, Data curation. **Jerzy Nawrocki:** Writing – review & editing, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank all participants of the study for sharing their experience and opinions.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jss.2022.111479.

References

Agrawal, R., Srikant, R., et al., 1994. Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, Vol. 1215. Citeseer, pp. 487–499.
Akoglu, H., 2018. User's guide to correlation coefficients. Turk. J. Emerg. Med. 18 (3), 91–93.
Alsaqaf, W., Daneva, M., Wieringa, R., 2019. Quality requirements challenges in the context of large-scale distributed agile: An empirical study. Inf. Softw. Technol. 110, 39–55.
Anderson, D., 2016. Essential Kanban Condensed. Blue Hole Press.
Charmaz, K., 2006. Constructing Grounded Theory: A Practical Guide Through Qualitative Analysis. Sage.
Cohn, M., 2010. Succeeding with Agile: Software Development using Scrum. Pearson Education.
Davis, N., 2013. Driving quality improvement and reducing technical debt with the definition of done. In: Proc. - Agil. 2013. pp. 164–168. http://dx.doi.org/10.1109/AGILE.2013.21.
Digital.ai, 2021. 15Th annual state of agile report. URL https://stateofagile.com.
European commision, 2020. Structural business statistics - small and medium-sized enterprises (SMEs). https://ec.europa.eu/eurostat/web/structural-business-statistics/structural-business-statistics/sme, last access February 20, 2020.
Hill, P.R., 2011. Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration. McGraw-Hill Education.
Hoda, R., Salleh, N., Grundy, J., 2018. The rise and evolution of agile software development. IEEE Softw. 35 (5), 58–63.

Igaki, H., Fukuyasu, N., Saiki, S., Matsumoto, S., Kusumoto, S., 2014. Quantitative assessment with using ticket driven development for teaching scrum framework. In: 36th Int. Conf. Softw. Eng. ICSE Companion 2014 - Proc.. pp. 372–381. http://dx.doi.org/10.1145/2591062.2591162.

ISBSG, 2020. Software project data. https://www.isbsg.org/software-project-data/, last access February 27, 2020.

Jakobsen, C.R., Johnson, K.A., 2008. Mature agile with a twist of CMMI. In: Agil. 2008. Agil. Conf.. IEEE, pp. 212–217.

Kasauli, R., Knauss, E., Nakatumba-Nabende, J., Kanagwa, B., 2020. Agile islands in a waterfall environment: Challenges and strategies in automotive. In: Proceedings of the Evaluation and Assessment in Software Engineering. pp. 31–40.

Kopczyńska, S., Nawrocki, J., Ochodek, M., 2018. An empirical study on catalog of non-functional requirement templates: Usefulness and maintenance issues. Inf. Softw. Technol. 103, 75–91.

Kopczyńska, S., Ochodek, M., Nawrocki, J., 2020. On importance of non-functional requirements in agile software projects—a survey. In: Integrating Research and Practice in Software Engineering. Springer, pp. 145–158.

Madan, S., 2019. Done understanding of the definition of done. https://www.scrum.org/resources/blog/done-understanding-definition-done, last access August 20, 2021.

Masood, Z., Hoda, R., Blincoe, K., 2020. How agile teams make self-assignment work: a grounded theory study. Empir. Softw. Eng. 25 (6), 4962–5005.

Matharu, G.S., Mishra, A., Singh, H., Upadhyay, P., 2015. Empirical study of agile software development methodologies: A comparative analysis. SIGSOFT Softw. Eng. Not. 40 (1), 1–6. http://dx.doi.org/10.1145/2693208.2693233.

Molléri, J.S., Petersen, K., Mendes, E., 2020. An empirically evaluated checklist for surveys in software engineering. Inf. Softw. Technol. 119, 106240.

Narayan, S., 2018. Products over projects. https://martinfowler.com/articles/products-over-projects.html, last accessed 10/03/2022.

Nurdiani, I., Börstler, J., Fricker, S., Petersen, K., Chatzipetrou, P., 2019. Understanding the order of agile practice introduction: Comparing agile maturity models and practitioners' experience. J. Syst. Softw. 156, 1–20.

Ochodek, M., Kopczyńska, S., 2018. Perceived importance of agile requirements engineering practices–a survey. J. Syst. Softw. 143, 29–43.

O'Connor, C.P., 2010. Letters from the edge of an agile transition. In: Proc. ACM Int. Conf. Companion Object Oriented Program. Syst. Lang. Appl. Companion, SPLASH '10. pp. 79–84. http://dx.doi.org/10.1145/1869542.1869557.

Paasivaara, M., Behm, B., Lassenius, C., Hallikainen, M., 2018. Large-scale agile transformation at ericsson: a case study. Empir. Softw. Eng. 23 (5), 2550–2596.

Povilaitis, S., 2014. Acceptance criteria. https://www.leadingagile.com/2014/09/acceptance-criteria, last access August 22, 2021.

Power, K., 2014a. Definition of ready: An experience report from teams at cisco. In: Agil. Process. Softw. Eng. Extrem. Program.. Springer, pp. 312–319.

Power, K., 2014b. Social contracts, simple rules and self-organization: A perspective on agile development. In: Agile Processes in Software Engineering and Extreeme Programming Conference. Springer.

Project Management Institute, 2017. Pulse of profession, success rates rise, transforming the high cost of low performance. https://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/pulse-of-the-profession-2017.pdf, last access March 24, 2021.

Rubin, K., 2013. Essential Scum: A Practical Guide to the Most Popular Agile Process. Pearson Education, Inc..

Saddington, P., 2012. Scaling agile product ownership through team alignment and optimization: A story of epic proportions. In: Proc. - 2012 Agil. Conf. Agil. 2012. IEEE, pp. 123–130. http://dx.doi.org/10.1109/Agile.2012.19.

Schwaber, K., Sutherland, J., 2020. The scrum guide. The definitive guide to scrum: The rules of the game. scrum.org.

Silva, A., Araújo, T., Nunes, J., Perkusich, M., Dilorenzo, E., Almeida, H., Perkusich, A., 2017. A systematic review on the use of definition of done on agile software development projects. In: ACM Int. Conf. Proceeding Ser., Vol. Part F1286. http://dx.doi.org/10.1145/3084226.3084262.

Silva, A., Dilorenzo, E., Perkusich, M., Santos, D., Almeida, H., Perkusich, A., 2018. Definition of done from academy to the industry: An exploratory survey. Int. J. Eng. Trends Technol. 58 (2), http://dx.doi.org/10.14445/22315381/ijett-v58p214.

Taipale, M., 2010. Huitale – a story of a finnish lean startup. In: International Conference on Lean Enterprise Software Systems.

Theobald, S., Schmitt, A., Diebold, P., 2019. Comparing scaling agile frameworks based on underlying practices. In: International Conference on Agile Software Development. Springer, pp. 88–96.

Vinson, N.G., Singer, J., 2008. A practical guide to ethical research involving humans. In: Guide to Advanced Empirical Software Engineering. Springer, pp. 229–256.

Viscardi, S., 2013. The Professional ScrumMaster's Handbook : A Collection of Tips, Tricks, and War Stories To Help the Professional ScrumMaster Break the Chains of Traditional Organization and Management. Packt Publishing, p. 314, URL https://www.packtpub.com/mapt/book/application_development/9781849688024.

Wiegers, K.E., Beatty, J., 2013. Software Requirements 3. Microsoft Press, USA.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A., 2012. Experimentation in Software Engineering. Springer Science & Business Media.

Sylwia Kopczyńska holds Ph.D. degree and works as assistant professor in the Institute of Computing Science at Poznan University of Technology. Her current research focuses on non-functional requirements engineering, software quality and measurement. She has been participating in national and international projects, mainly as analyst, developer in the domains of software development, insurance, transportation, travel, pharmacy.

Miroslaw Ochodek holds Ph.D. degree and works as an assistant professor in the Institute of Computing Science at the Poznan University of Technology. His main research interests lie in the fields of requirements engineering, software metrics, functional size measurement, and software effort estimation.

Jaktb Piechowiak graduated from Poznan University of Technology with a degree in Computer Science, Sofware Engineering. He is a practitioner interested in agile software development approaches and Java programming. Currently, he works as Java Developer at Capgemini.

Jerzy Nawrocki received his M.Sc. degree (1980), Ph.D. degree (1984), and Dr. hab. degree (1994)—all in informatics and all from the Poznan University of Technology (PUT), Poznan, Poland. His research interests concern Sofware Engineering and Project Management. Currently he is the Dean of the Faculty of Computing at PUT, a Vice-Chair of IFIP Technical Committee 2 Sofware Theory and Practice (since 011), and an IFIP Board Member (term 013–016).