Definition of Done from Academy to the Industry: An Exploratory Survey

Ana Silva§, Ednaldo Dilorenzo¶, Mirko Perkusich¶, Danilo Santos¶, Hyggo Almeida¶, Angelo Perkusich¶

§*Federal Institute of Paraíba*
Monteiro, Paraíba, Brazil, 58500-000
anasilva.ifpb@gmail.com

¶*Federal University of Campina Grande*
Campina Grande, Paraíba, Brazil, 58429-140
{ednaldofilho}@copin.ufcg.edu.br, {mirko.perkusich, danilo.santos, hyggo, perkusic}@embedded.ufcg.edu.br

Abstract — *Definition of Done (DoD) assures a balance between short-term delivery of features and long-term product quality. The state of the art of this area was documented in a Systematic Literature Review (SLR), which found 8 papers. This study reports findings on the usage of DoD based on the SLR results, focusing on aspects such as which criteria are used and the process in which they are defined and assessed. A survey was carried out using online questionnaires answered by practitioners. Data was collected from agile practitioners from 16 countries and different team sizes, and the main findings are: 1) Quality management is the most popular type of criteria; (2) DoD are not defined following a standard; 3) Most projects use a single level DoD; 4) DoD is emergent during a project; 5) The criteria are visible; 6) Most projects assess DoD manually; 7) There is no empirical evidence of its effectiveness; (8) Most challenges are related to the risk of over effort and reaching agreement among stakeholders; and (9) a new criteria were reported, deploy. The main implication for research is a need for empirical studies exploring its use. For the industry, the results can assist the definition of their own.*

Keywords — *Survey, Definition of Done, Agile Software Development, Software Development.*

I. Introduction

To achieve a successful use of Agile Software Development (ASD), it is necessary to adopt the correct practices given the project's context. Among such practices, Definition of Done (DoD) – which is part of the Scrum Framework [6] – consists of a set of criteria to define if a deliverable is done: the minimal restrictions that must be fulfilled for a product to be released [5]. According to Sutherland and Schwaber [6], DoD promotes transparency between the stakeholders on the meaning of completing work. According to Williams [7], in which data collected from 326 practitioners were analysed, it is the most popular agile practice along with short iterations and continuous integration.

In our previous work, we performed a Systematic Literature Review (SLR) to explore the state of the art in DoD. As a result, 8 studies reported the use of this practice, where 5 studies reported the use of only 1 level of done. At most, 4 levels are used (i.e., Story, Sprint, Release and Project), in which each level contains a set of criteria. Each criterion can be related to a software development or management practice, conformance with external standards or artefacts (e.g., requirements, design and product) and non-functional requirements. For instance, examples of criteria are: check if code was peer reviewed and if performance tests passed given a set of metrics and targets.

Sixty-two (62) criteria has been identified and over 50% were related to regulatory compliance, which is the case of ASD projects in the context of markets with strict compliance processes such as medical and space industry. On the other hand, the most frequent criteria are related to quality procedures, such as unit test, peer code review and acceptance test. Due to its focus on fully refereed empirically-based research, the previous SLR did not include works that were not peer reviewed, such as white papers, technical reports, and practitioners' opinion in articles/forums/blogs.

This paper, therefore, complements the findings of [5] by gathering evidences, discussing and providing a combined and detailed understanding regarding the usage of DoD in practice based on practitioners' experience.

This paper is structured as follows: Section II describes the methodology used in this study; Section III presents our findings; Section IV discusses the results; Section V shows our conclusions and recommendations for future research.

II. Methodology

Considering that our previous work [5] has presented the state of the art in DoD, we have deepened our investigation with a study with practitioners. For this purpose, we have defined the following research questions:

- **RQ1:** What are the done criteria used in agile software development projects?
- **RQ2:** Do organizations have a standard process or a core set for defining the done criteria? If yes, what process is used?
- **RQ3:** Is the definition of "done" divided into levels? If yes, which levels are used?
- **RQ4:** Is the DoD emergent during a project?
- **RQ5:** Are the criteria of Done explicit (i.e., documented) or tacit?
- **RQ6:** How are the criteria of Done assessed during the project execution?
- **RQ7:** Are there evidences of the effectiveness of using the DoD? If so, how was this verified by the team?
- **RQ8:** What are the challenges in using DoD? What are their consequences?

To answer our research questions, we employed a survey to gather data from practitioners using as an instrument an on- line questionnaire. There are several advantages of performing an on-line survey, such as:

- low cost of questionnaire distribution and to code data;
- short turnaround time;
- reaches respondents all around the world;
- offers a mean to survey many individuals;
- may increase respondents' motivation to participate by providing an interactive survey process;
- may reduce errors from transcription and coding.

To perform the survey, questions need to be formulated, pretested (i.e., piloted), released, and any extra information that can avoid misunderstandings should be placed [4]. The questionnaire was organized into two parts: part 1 was composed of questions to gather demographic data, while part 2 of questions to answer the research questions. In part 2, we created at least one question in the questionnaire for each research question, because there could be closed and open-ended questions. The answers to open-ended questions were analyzed using thematic analysis [1].

Some risks associated to Internet-based survey are: effect of self-selection; multiple responses from the same respondent; and difficulty in reporting response rate [9]. A survey is worthless if the answers provided by the respondents are not meaningful. Response distortions in on-line surveys are mostly attributed to errors caused by the instrument or by the interviewee [2]. To minimize these threats, we created a pilot questionnaire using an online form and delivered the first version of the study to 6 practitioners with experience managing agile projects and using DoD. After responding the survey, we sent them set of questions to assess the survey in terms of: understandability, bias, sensitivity and completeness. As a result, we modified the sequence, wording and response options of a few questions. These answers were not used as result of the survey.

Afterwards, we released the questionnaire, which is still available1, on professional social networks, discussion groups and with industry partners. Table I shows Part 1 of the questionnaire and Table II shows Part 2 of the questionnaire.

TABLE I: Demographic questions of the survey.

Id	Question
1	What is the size of your company (in full-time staff)? If you work for,a multinational, you must consider all employees of the company, not only your unit.
2	What is the size of the team allocated for the given project?
3	What is the type of system developed in the given project?
4	In which country is the project being executed? In case of global distributed teams, you may select more than one.
5	Is the project developed in a geographically distributed manner?
6	What is your main role in the company?
7	How many years of experience do you have in agile software development?
8	Which agile methods are/were used in the project?
9	In case of an eventual failure of the system, which could be the consequences?

TABLE II: Research questions of the survey

Id	Question
10	Does your organization have a standard process for defining the Definition of "Done"?
11	What is the process to define a definition of "done" in your organization? If possible, describe it.
12	In your company, is there a core set of definition of "done" (or definition of "done" types) for every project?
13	Is the definition of "done" divided into levels (user stories, sprint, release, etc.)?
14	If you answered yes to the previous question, which levels of "done" are used?
15	What are the "done" criteria used? If the definition of, "done" is divide into levels, specify which "done" criteria are used at each level.
16	Did the Definition of "done" criteria change/emerge during development stage of the project? If so, how and why?
17	Are the criteria of Done explicit (i.e., documented) or tacit?
18	If you answered "Explicit" to the preceding question, how is the criteria of "done" made available to the team?
19	How are the criteria of Done assessed during the project execution? You may select more than one option in case you have different assessment procedures for different criteria.
20	Please, describe the procedure assessment for each of the criteria of Done used in your project.
21	Has the effectiveness of using the definition of "done" (levels and criteria) during a software development project been proven? If so, how was this verified by the team?
22	What are the challenges in creating a definition of "done"?
23	What are the challenges in assessing the "done" criteria during the project execution?

24	Did any of the challenges above prevent you from using the practice in other projects?
25	If you answered "yes" to the preceding question, explain why it happened.

A. Threats to validity

This section discusses validity threats using the four types of threats suggested in [8].

- **Construct validity** is concerned with issues that may arise due to improper design of the survey instrument, which then may not be measuring properly what it is supposed to measure. We minimized this threat with the pilot study with 6 practitioners.
- **Conclusion validity** is related with the possibility of reaching incorrect conclusions about association in observations due to errors such as use of inadequate statistical tests or measures. In this study, we only used frequencies and percentages to identify common patterns or practices to point out potential areas or relationships for future research efforts. In addition, we only considered complete responses in our analysis.
- **Internal validity** is concerned with issues, such as confounding factors or irrelevant respondents, which could introduce a systematic error or bias in the study results. To mitigate this threat, we asked the respondents regarding their experience with ASD and, in the introduction of the survey, made clear that only evidence from industry projects should be entered.
- **External validity** refers to the extent to which findings in a study are applicable outside of the study context. Since the sample used herein was a convenience sample and its size are 20, our results are only generalizable to those agile teams and companies that share similar characteristics to our survey respondents' teams and companies. However, we advertised the questionnaire on several communities and obtained a quite heterogeneous sample in terms of company size, experience and country.

III. Results

We received answers from 20 agile professionals spread over 16 countries, in which a professional might work in a global project (i.e., executed in multiple countries). In Figure 1, we present the distribution of respondents per country.

Most of the respondents work on medium sized or very large companies as Figure 2 shows. Furthermore, 70% work on teams with between 6 to 10 members, 20%, more than 20 members and 10% between 1 and 5 members. 60% work on geographically distributed teams, in which 30% is distributed nationally and 30%, globally. Most of the respondents worked on projects to develop web applications, as Figure 3 shows. Regarding the agile methods, in which more than one could be used in a single project, 80% reported to use Scrum; 50%, Kanban; 40%, Extreme Programming; 20%, ScrumBan; and 20%, Lean.

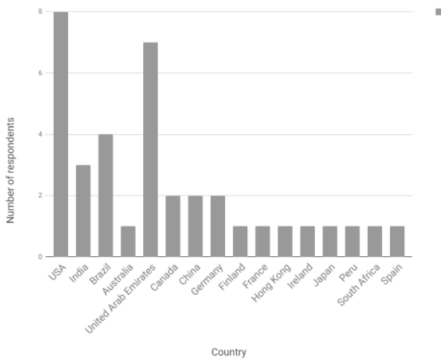

Fig. 1: Number of respondents of the survey per country.

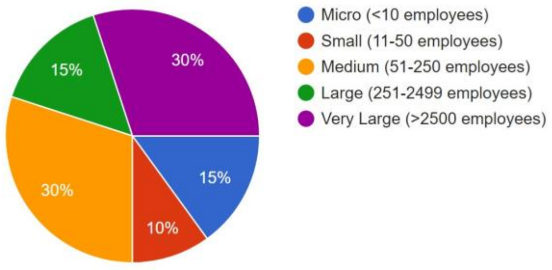

Fig. 2: Size of the companies of the respondents.

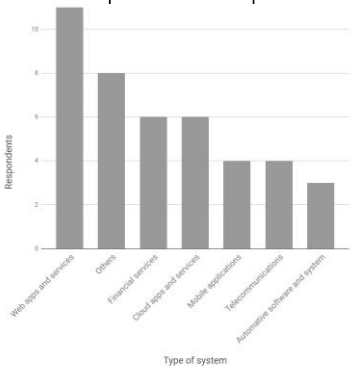

Fig. 3: Type of the systems developed by the projects of the respondents.

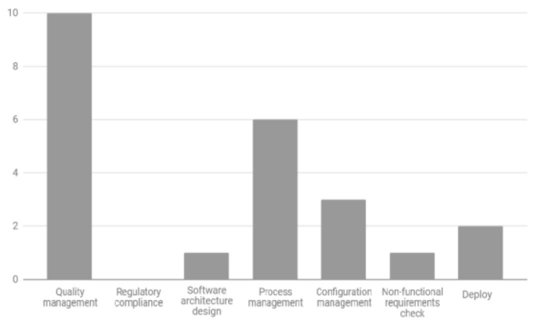

Fig. 4: Frequency of category of done criteria

All the respondents perform managerial roles, including Scrum Master (40%), project manager (30%), product manager (15%) and agile coach (15%). Besides, 40% of the respondents have more than 10 years of experience in agile development, while 30% between 5 and 10 years of experience, 15% have between 3 and 5 years of experience, and 15% have between 1 and 3 years of experience.

A. Done Criteria

Question: What are the done criteria used in agile software development projects?

To report the identified *done criteria*, we initially used the same categories of [5]: Quality management, regulatory compliance, software architecture design, process management, configuration management, non-functional requirements. During the study, we identified a new category: deploy. Table III presents the *done criteria* reported by the respondents, exactly as they answered, grouped by categories and identified by the respondent id, where R1 means Respondent 1 and so on. Figure 4 presents the frequency of each category per number of respondents.

B. Process to define DoD

Question: Do organizations have a standard process or a core set for defining the done criteria? If yes, what process is used?

From responses, 80% of the respondents reported that, in their companies, they define the DoD for the projects given its context, independent of a core set of done criteria. Three (3) respondents claimed that in their project there is not a clear concept of "done" and 1 answered that in his company the DoD for the projects are defined based on an organizational standard. In most cases, the teams are responsible to define the DoD for the project through brainstorming sessions. In what follows, we present quotes from the respondents, followed by an identifier and the role.

"The tribe of 3 squads created a joint DoD, because they all work on the same code base." P6, Scrum Master.

"We have a team discussion, write down our thoughts, and then we typically visualize it on a poster and keep it next to our sprint board. Photos thereof are uploaded onto Slack and we take it along to sessions and reference it there." P9, Scrum Master.

"Defined by development team. Enough to ensure work is maintainable long term, understood by all the team, installable, meets quality standards constantly evolving and has been peer reviewed." P10, Scrum Master.

TABLE III: DoD criteria by categories.

Category	**Answers**
Quality Management	R1. Code and tests are reviewed, or pair programmed, the code is non-embarrassing, code has automatic tests and tests pass, exploratory tests have been done. R2. Reviewed by collaboration, pair programming or peer review, build good, Product Owner accepts story function, Unit tests passed, smoke tests pass, systems tests failures addressed. R6. Unit tested, all functional tests passed, automated regression testing written and tested. R8. Anything we've agreed with stakeholders, that we're at risk of forgetting. R9. Peer code review done, builds pass locally and on server, acceptance criteria met, critical bugs fixed, unit tests passed, with 100% path coverage, functional tests executed, tested and approved by Product Owner. R15. Product Owner and Sponsors are satisfied. R16. unit tests in place, acceptance tests. R17. quality criteria approved (code metrics), quality criteria approved (tests), UAT approved in customer environment. R18. Typically, both team and client consider results done after simple testing the system presents minor or no mis- takes. R19. I use Acceptance criteria as DoD.
Software architecture design	R1. APIs are reviewed, component/class designs are reviewed.
Non-functional requirements check	R15. Security and Load Tests are deemed satisfactory.
Process management	R1. Internal and external documentation is updated, acceptance criteria are met, demo is prepared. R2. All JIRA tasks closed. R6. Developed. R9. Swagger doc created for each service, Data Dictionary updated, bugs prioritized by the Product Owner. R16. Documentation. R17. Functional requirements implemented, knowledge transfer done.
Configuration management	R1. Code is submitted to master. R2. Code checked in. R16. Continuous integration done.
Deploy	R4. If delivery to customer not possible then a staging area as close to customer environment as possible. R9. Deployed to QA (our project is not deploying to Prod at the end of each sprint).

C. Levels of DoD

Question: Is the definition of "done" divided into levels? If yes, which levels are used?

From responses, 75% of the respondents use a single level of DoD. Out of the 5 respondents that use multiple levels of DoD, 2 of them use 3 levels: story, sprint and release; 3 of them use 2 levels: story and release.

D. DoD Evolution During Project

Question: Is the DoD emergent during a project?

From responses, 70% of the respondents claimed that the DoD changed during the project. The most common reasons to change the DoD are increase in team maturity and volatile requirements. As long the teams develop the product and know the customer, more they learn about the best process to be executed to guarantee quality and customer satisfaction. Therefore, there is a need for the teams and their process to be adaptable to the learning.

"Team started with a set of criteria that looked reasonable. The Done criteria where refined in retrospectives periodically by asking ourselves whether they help or keep us from getting work done." P1, Product Manager.

Volatile requirements, which is a characteristic of ASD, also causes changes to the DoD. Changes in the customer environment might cause change in requirements and quality criteria.

"Yes. In some contexts, the client matures his/her personal acceptance criteria for a done status, thus adjustments are necessary." P18, Project Manager.

E. DoD Documentation

Question: Are the criteria of Done explicit (i.e., documented) or tacit?

From responses, 80% of the respondents claimed that in their companies the DoD criteria are explicitly documented. Among the ones who document DoD, 58.8% use wallboard to make it accessible to the team and 23.5% use electronic tools such as Wiki or project management tools.

F. DoD Assessment

Question: How are the criteria of Done assessed during the project execution?

Assessing the DoD is a key activity to assure that the process is being followed. 30% (6) of the respondents assess DoD only manually, 15% (3) use only tools for this purpose while 5% (1) use only other methods. Most respondents – 50% (10) – use a mix of criteria (tools, manual, other, etc.) to assess DoD. 25% (5) respondents reported that they use checklists either made by developers, quality assessment team, or Product Owner to verify given artifacts.

For instance, P1, a product manager, reported that the DoD is made visible on the Scrum board as a checklist. Before a story is moved to "Done", it is consulted manually by the team members during the Daily Scrum.

Another form of DoD assessment is through use of manually or automated tested features to ensure the stories have an acceptable quality. Only 1 respondent claimed to use metrics to verify DoD.

"Development team uses a checklist (implementation finished, review done, static tests passed, etc.), test team measure product quality metrics (defects, pass rate, performance, etc.), project manager control release DoD evaluation during project delivery." P17, Project manager, software architect and quality manager.

"Documentation is assessed manually. Criteria related to tests are verified through tools that executes daily on continuous integration process" P16, Software architect, team leader and technical leader.

Furthermore, 8 of the respondents interpreted DoD assessment as code and product assessment. Therefore, for product assessment, they claimed to have the customer to check the quality of the delivered product. 2 respondents claimed to use peer code review to assess the code.

G. Effectiveness of DoD

Question: Are there evidences of the effectiveness of using the DoD? If so, how was this verified by the team?

All respondents described they believe the DoD criteria aid the project achieve better results, based on the customer's positive feedback, the team feeling, and the negative impact on final product which did not follow the DoD criteria before being deployed in production environment. However, none of the respondents described a sound empirical method applied to verify their belief.

"It is accepted without proof, that's why the criteria are adapted to new learnings." P1, Product Manager.

"When there is an unexpected problem after release, we frequently find that some definition was violated." P2, Agile coach.

"Yes, long term productivity morale and ease of estimation. Fewer installation related defects and rework." P10, Scrum Master.

"If the team thinks it helps they do it." P8, non-defined role.

H. Challenges Using DoD

Question: What are the challenges in using DoD? What are their consequences?

Five (5) respondents reported that the biggest challenge to define DoD is the risk of over effort, which might be caused by over documentation, tests and complex deploy procedure.

"Get to done on production-like environment proves hard in finance" P14, Agile coach.

"Naive/amateur product owners (or stakeholders) do not understand that software is a complex product subject to failure in several levels and state and agreement about this

sometimes is a hard task. More than that, there is a complexity on finding the correct trade-off between quality needs and the other variable of the iron triangle. Basically, the done criteria may vary from a naive review of what was executed against the acceptance criteria of user stories. But can also be based on quality statements and metrics that can be hard to extract from development process, increasing the need for documentation and slowing the dev. process. That demands negotiations with clients." P18, Project manager.

Five (5) respondents claimed that, sometimes, it might be challenging to get into an agreement with the team and customers regarding the DoD. 3 reported that the unknown causes of uncertainty regarding the fit in which the DoD is enough to satisfy the customer.

"To understand correctly future procedures and criteria that will be applied for product and project approval." P17, Project manager, software architect and quality manager.

On the other hand, 4 respondents claimed that there are no challenges to define DoD:

"None for me, nothing is unachievable, and I have 31 years (14 Agile) of doing this." P15, Agile coach.

Regarding the assessment of DoD, the only challenge reported is the effort on meetings and artifact evaluations and the risk of human error. So, automating the DoD evaluation is necessary if the quality criteria regarding the product or process is high.

IV. Discussion

Most of the respondents of the survey are experienced agile practitioners and reported projects using Scrum. DoD criteria elicited on the survey (see Figure 4) show that the most frequent category is quality management, validating the result stated in our last work [5]. This is expected since the goal of DoD is it to define minimal restrictions that must be fulfilled for a product to be released [5], which are usually defined as quality criteria.

Other similarities between both works are regarding criteria related to configuration management, software architecture and non-functional requirements check. Regarding configuration management, the focus is mainly to assess if the commit policies are being followed. Software architecture check is not common on both cases but focuses on assessing documentation regarding the software architecture and design. Using DoD to check non-functional requirements were presented in both results and presents an interesting topic for discussion.

Some non-functional requirements such as internationalization might be associated with all Product Backlog items (PBI), if it is necessary to implement frontends to attend them. On the other hand, other non-functional requirements such as security might only be useful for a single PBI. For instance, cryptography of user password or credit card information might only be associated with one PBI. Therefore, we hypothesize that some non-functional requirements are appropriate to be handled through DoD; others, could be implemented as a PBI.

On the other hand, regulatory compliance was a category identified through the literature review but not reported on the survey. Furthermore, process management-related criteria such as documentation and update of tools were more frequent in the survey.

The survey identified a new category: Deploy, which relates to deploy-related conditions that the team must adhere. For instance, one respondent claimed that a delivery was only done whenever the code was deployed to the staging (i.e. production-like) environment. In Scrum, having the product deployed in a production-like environment during Sprint Reviews is essential if there is a goal to release versions of software every sprint. On the other hand, this is not the case for all projects. For instance, in the case of innovation projects, during initial phases, sprints might produce prototype versions of the software to validate the idea and better understand the problem.

The survey also explored other topics such as the use of levels of DoD, DoD evolution during a project, documentation, assessment and challenges. Regarding the use of levels, most of the respondents use a single level of DoD (see Section III-C). We did not find any correlation between the characteristics of the organization or company and the number of levels of DoD used. All the projects with multiple levels of DoD are distributed globally or locally, but, respondents from 7 distributed projects argued that only use a single level of DoD. Therefore, we believe that there is a need for research to understand the motivations and consequences of using multiple levels of DoD.

Regarding the evolution of DoD during a project (see Section III-D), as expected most respondents claimed that it evolves during the project given the team's learning and the volatile environment, which is natural for ASD. Most teams use explicit communication of DoD through documentation such as wallboard, wiki and project management tools (see Section III-E) what is reasonable, main for projects geographically distributed.

As reported in the results of the SLR (see Section III-F), practitioners use checklists and metrics to assess the DoD criteria. Even though most respondents reported that they do not use a standard DoD as a guideline to define the DoD for projects, we believe that there is an opportunity for improvement in agile companies by defining a core set of DoD criteria, which could be the minimum that each project should adhere. We believe that this reasoning is not against agile principles of having a self-managed team, such as we inferred by some of the responses on the survey. On the other hand, the standards should be knowledge sharing activity using assets of the company, namely checklists, activities and metrics program, which should be continuously evolved through the use of lessons learned for each project. For mature and experienced teams, it might not be necessary to have the assistance of the assets, but for others, it might be valuable.

Regarding the challenges of applying DoD (see Section III-H), most of them are related to the risk of over effort and having agreement between the team and the customer. These challenges can be minimized if there is a recommendation of DoD criteria for a given projects' context which, as reported on the results of the SLR, the literature is scarce. The effort of assessing DoD in development teams was also reported since DoD can significantly increase the quality criteria that enables a software product to go under production. In this sense, we believe the goal of companies must be to automate the entire assessment with approaches such as deployment pipelines [3].

As reported and concluded on the results of the SLR, there is a gap in the state of the art and practice regarding empirical data to show the effectiveness of methods to apply DoD. The survey has shown that the effectiveness of DoD is based on the feeling of the respondents and we believe effort should be made to evaluate the effectiveness of this practice through empirically sound methods.

V. Conclusion

This paper reports a survey in the field of DoD in ASD with practitioners. The conducted work is the first empirical study that tries to map the usage of DoD in practice. The research questions aimed at generating a comprehensive overview and the answers to these questions deliver potential benefits for both research and practitioners communities. Researchers may use the study results as reference and starting point for their own research projects, and practitioners may use the results as a reference to apply DoD to their companies.

We received answers from 20 practitioners from 16 countries, where we identified a variety of done criteria used in agile software development. For example, some projects use a multilevel approach to manage DoD, which include story, sprint, release or project, however, we did not find any correlation between the number of levels and the project context. The results of the previous SLR [5] and this survey agreed that quality management is the most popular type of criteria. Also, they agreed that using configuration management, software architecture and non-functional requirements checks criteria are used by companies but are not very popular. On the other hand, even though regulatory compliance was identified during the SLR, it was not reported in the survey. As a new finding, deploy criteria were reported in the survey, but not reported in the SLR.

We found that DoD is used even in agile projects that do not use Scrum, which bring us evidences about its importance. Checklists and metrics shown to be popular mechanisms for DoD assessment in the survey.

Moreover, based on the results of this study, we recommend a strong need to publish more works presenting how DoD is applied in agile projects, and how to conduct empirical studies to assess the results of applying this practice. For future work, we intend to execute case studies in companies to deepen our understanding of the reasoning and efficiency of applying DoD.

References

[1] D. S. Cruzes and T. Dybå. Research synthesis in software engineering: A tertiary study. *Information and Software Technology*, 53(5):440 – 455, 2011. Special Section on Best Papers from XP2010.

[2] L. Graf. Assessing internet questionnaires: The online pretest lab. *Online social sciences*, pages 49–68, 2002.

[3] J. Humble and D. Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. 2010.

[4] M. Perkusich, A. Perkusich, and H. Almeida. Using survey and weighted functions to generate node probability tables for Bayesian networks. In *Proceedings of BRICS-CCI 2013*, 2013.

[5] A. Silva, T. Araújo, J. Nunes, M. Perkusich, E. Dilorenzo, H. Almeida, and A. Perkusich. A systematic review on the use of definition of done on agile software development projects. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, EASE'17*, pages 364–373, New York, NY, USA, 2017. ACM.

[6] J. Sutherland and K. Schwaber. The scrum guide. [Online]. Available: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf, 2013. Accessed: 2016-03-02.

[7] L. Williams. What agile teams think of agile principles. *Commun. ACM*, 55(4):71–76, Apr. 2012.

[8] C. Wohlin, P. Runeson, M. Ho¨st, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[9] Y. Zhang. Using the internet for survey research: A case study. *Journal of the American Society for Information Science*, 51(1):57–68, 2000.