4-15-2019

Use of Software Process in Research Software Development:A Survey

Nasir U. Eisty
University of Alabama - Tuscaloosa

George K. Thiruvathukal
Loyola University Chicago, gkt@cs.luc.edu

Jeffrey C. Carver
University of Alabama - Tuscaloosa

Recommended Citation

Nasir U. Eisty, George K. Thiruvathukal, and Jeffrey C. Carver, Use of Software Process in Research Software Development: A Survey, Proceedings of the Evaluation and Assessment on Software Engineering 2019.

Use of Software Process in Research Software Development: A Survey

Nasir U. Eisty
University of Alabama
Tuscaloosa, AL
neisty@crimson.ua.edu

George K. Thiruvathukal
Loyola University Chicago
Chicago, IL
gkt@cs.luc.edu

Jeffrey C. Carver
University of Alabama
Tuscaloosa, AL
carver@cs.ua.edu

ABSTRACT

Background: Developers face challenges in building high-quality research software due to its inherent complexity. These challenges can reduce the confidence users have in the quality of the result produced by the software. Use of a defined software development process, which divides the development into distinct phases, results in improved design, more trustworthy results, and better project management. *Aims*: This paper focuses on gaining a better understanding of the use of software development process for research software. *Method*: We surveyed research software developers to collect information about their use of software development processes. We analyze whether and demographic factors influence the respondents' use of and perceived value in defined process. *Results*: Based on 98 responses, research software developers appear to follow a defined software development process at least some of the time. The respondents also have a strong positive perception about the value of following processes. *Conclusions*: To produce high-quality and reliable research software, which is critical for many research domains, research software developers must follow a proper software development process. The results indicate a positive perception of value about using defined development processes that should lead to both short-term benefits through improved results and long-term benefits through more maintainable software.

CCS CONCEPTS

• **General and reference → Surveys and overviews**; • **Software and its engineering → Software development methods**.

KEYWORDS

software process; research software; survey

ACM Reference Format:
Nasir U. Eisty, George K. Thiruvathukal, and Jeffrey C. Carver. 2019. Use of Software Process in Research Software Development: A Survey. In *Evaluation and Assessment in Software Engineering (EASE '19), April 15–17, 2019, Copenhagen, Denmark*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3319008.3319351

EASE '19, April 15–17, 2019, Copenhagen, Denmark

ACM ISBN 978-1-4503-7145-2/19/04…$15.00
https://doi.org/10.1145/3319008.3319351

1 INTRODUCTION

Researchers in a number of scientific, engineering, business, and humanities domains increasingly develop and/or use software to conduct or support their research. These researchers draw insights and make critical decisions based, at least partially, upon the software's output. This software, referred to as *research software*, consists of tools, libraries, and end-user software.

Because of the decisions made based upon the research software, the correctness of its design and implementation is of utmost importance. Low-quality software is likely to produce less trustworthy results and may lead to incorrect research conclusions or engineering/design decisions. The characteristics of research software and of the research software developer lead them to be more concerned about the underlying scientific goal than about the use of appropriate software engineering (SE) practices.

While our experience with research software suggests that its developers place a lower priority on SE practice, in the software industry at large, software process is of critical importance in building reliable software, especially in complex domains and/or mission-critical environments. Previous work has shown that research software teams may follow agile-like processes, even if they are unaware of those processes [2, 8]. Therefore, the goal of this work is to *gain a better understanding of the use of software development process for research software in support of the development process.* To gather the information on software process, we developed and distributed a survey to research software developers. The survey provided respondents an opportunity to provide feedback on the use of software process in their respective projects. The survey reported in this paper is part of a larger survey on the use of software metrics and software process in research software. We have previously published the results related to the use of software metrics [6]. This paper focuses only on the results related to software process.

The primary contributions of this paper are:

- An overview of how research software developers use and perceive the importance of software process;
- Identification of the software processes preferred by research software developers;
- The perceived prevalence of software process and whether it is useful to research software developers;

The remainder of this paper is organized as follows. In Section 2 we describe previous work to motivate a series of research questions explored in this study. In Section 3 we explain the survey design. In Section 4 we provide the detailed survey results. In Section 5 we discuss the validity threats. In Section 6 we draw conclusions.

2 RESEARCH QUESTIONS

In this section, we define our research questions based upon a discussion of the related work. These research questions drive the survey design.

The literature shows that researchers in a number of science and engineering domains increasingly use software development processes. The reports from these domains, which include physics [7], astrophysics [5], and biomedical imaging [4], reveal a range of goals for software process. The goals include developing modular and extensible software to developing software that is open for user contributions. Research software developers in the biomedical imaging domain specifically use software process to help improve interactions between domain scientists and developers.

At least one large-scale effort highlights the continued value of traditional software process for research software. The ASCI project, a project aimed at reducing the nuclear stockpile through simulation and modeling, made use of traditional software development processes. The ASCI project found that techniques associated with traditional process to be of value, including, but are not limited to, estimation, resource management, team expertise/skill sets, risk analysis, working with stakeholders, training, and verification and validation [10].

These observations lead to the first three research questions:

- ***RQ1:*** *To what extent do research software developers follow software process?*
- ***RQ2:*** *What value do research software developers place on software process?*
- ***RQ3:*** *Is there any relationship between the value placed on software process (RQ2) and the use of software process (RQ1)?*

A systematic literature review found that many research teams use agile approaches and variants thereof, even if they are unaware [8]. A review of *agile practices* [11] in scientific software development mapped the use of prominent agile reference models (e.g. Scrum and eXtreme Programming, a.k.a. XP, etc.). A key finding of this mapping study is that scientific software development projects not only adopt agile practices but also perceive their testing to be better than average.

Therefore, to better understand the types of software processes research software teams use the next research question is:

- ***RQ4:*** *Which software development processes do research software teams use?*

Research software developers will benefit from less intrusive processes [9]. To determine whether a particular software process is effective, the team must conduct measurement and analysis. The authors of this study propose collecting data for measuring process effectiveness to decide what practices might be most effective. Although there has been little follow-up to validate this approach, we find it positive that a research software team believe in the importance of measuring process with an eye towards improving it. Therefore, the next research question is:

- ***RQ5:*** *To what extent do research software teams use software process metrics?*

Lastly, a survey of research practitioners revealed some interesting findings about *software engineering knowledge* within the research community [2]:

- Most research practitioners have little formal SE training and tend to be self-taught.
- One-third of the respondents thought that overall the research communities' SE skills were not adequate.
- Familiarity of SE methods was higher than the use of those methods.
- Code reviews and agile methods were rarely used, suggesting a lack of collaborative development practices.
- The knowledge and perceived relevance of agile methods were low.

There is some inconsistency in the level of knowledge and understanding research software developers have about software development process. Because research software developers often have different backgrounds than traditional software engineers, various demographics could affect their level of knowledge about SE. To better understand whether these factors also affect research software developers, we pose the following research question:

- ***RQ6****: To what extent do the demographics of the respondents affect their perception and use of software process?*

3 SURVEY DESIGN

The results in this paper are part of a larger survey about the use of SE practices and metrics in the development of research software. In a prior paper [6], we reported results regarding how research software developers use (or do not use) software metrics and code metrics. In this paper, we analyze the questions related to use of software process. Therefore, the survey design process described here is the same as in our prior paper.

Using the research questions defined in Section 2, we enumerated a series of survey questions. Figure 1 shows the portion of the survey that we include in this analysis. We took care in writing the questions to ensure their wording did not bias the respondents. For example, in DQ1 we provided examples for each of the project types to ensure a common understanding from the survey respondents. We grouped the survey questions by topics to help focus the respondents' answers.

To reach a broad audience of research software domains we used three solicitation methods. First, we sent the survey invitation to a series of mailing lists that target developers and users of mathematical, science, and engineering software. Those mailing lists included: hpc-announce@mcs.anl.gov (a mailing list based at Argonne National Laboratory that targets researchers who use high-performance computing in their work), the PI list for the US National Science Foundation SI2 (Software Infrastructure for Sustained Innovation) PI mailing list, and Carver's list of previous participants in the SE4Science workshop series (http://www.SE4Science.org/workshops). Second, a collaborator sent the survey to the mailing list of research software developers in the UK. Third, we advertised the survey in a column in *Computing in Science & Engineering* [3] where many research software developers/practitioners would be likely to see it. In both cases, we also asked people to forward the survey invitation within their own networks. As a result of our solicitation approach, we are not able to estimate the number of people who received the invitation.

Figure 1: Survey Questions

Demographic Questions

DQ1 Which of the following best describes your project? [Scientific Computing Software, Computer Science Software, General Application Software, Other]

DQ2 How many FTEs of developers are currently on your project? [free-response]

DQ3 Which best describes your role on the project? [Developer, Architect, Manager, Executive, Other]

DQ4 Which best describes the current development stage of your project? [Planning/Requirements Gathering, Initial Development/Prototyping, Active Development/Unreleased Software, Active Development/Released Software, Maintenance/No New Development Planned, Other]

Process Questions

PQ1 How often does your team follow a defined software development process? [Never, Rarely, Sometimes, Most of the Time, Always]

PQ2 How often do you personally follow a defined software development process? [Never, Rarely, Sometimes, Most of the Time, Always]

PQ3 How much value do you see personally in following a defined software development process? [Very Little, Little, Moderate, High, Very High]

PQ4 How often does your team use each of the following software development processes? [Agile, Rapid Application Development, Ad Hoc, Waterfall model, Spiral model, Use-Case Methodology]

PQ5 How often does your team use process metrics to evaluate the effectiveness of the software development processes used on your project? [Never, Rarely, Sometimes, Most of the Time, Always]

4 RESULTS

We received 98 responses from people working on research software projects. We included a response in the analysis if the respondent answered at least 90% of the questions. Note that due to the different questions analyzed, the total number of respondents differs between this analysis and the analysis in the previous paper on this survey [6]. In this section, we first provide an overview of the demographic information about the sample. Then, we report the data for each of the research questions defined in Section 2. Throughout this section, the survey question numbers refer to the numbers in Figure 1.

Demographics

We provide a brief explanation of why the demographic is relevant for each of the demographics and any implications the demographic has for the survey analysis. In addition to providing an overall characterization of the sample, two of the four demographics (project size and project stage) factor into the analysis of RQ6.

Project Types. To ensure our sample captured the target population, question DQ1 gathers the type of the respondent's project. While Figure 1 lists the possible answers to the question, the full survey provided examples for each answer to clarify meaning. More than 90% of the respondents answered *Scientific Computing Software* for this question. This result indicates that the sample did come from our target population of research software developers.

Project Size. In Figure 2, we show the distribution of responses to DQ2 about the number of full-time equivalent (FTEs) staff currently on the the respondent's project. The left-ward skew of the results shows that our sample works mostly on smaller teams.

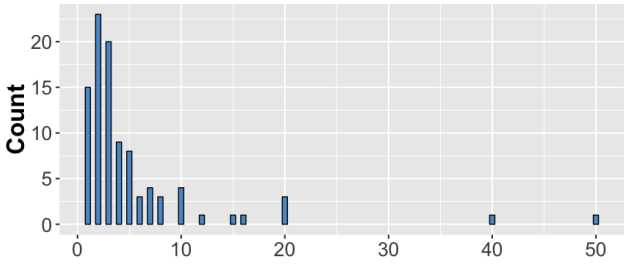

Figure 2: Number of Developers

Project Role. We asked in question DQ3 about the developer's role(s) because the role could affect his/her perception of software process. The results in Figure 3 show that the respondents were skewed more towards technical roles (developers and architects) than towards non-technical roles (managers, executives, and others). Note that because respondents could choose more than one role, the total in the figure is larger than the total number of respondents.

Project Development Stage. Question DQ4 lists the possible answer choices for project stage. In Figure 4, we show the project stage of the projects represented by the respondents. The majority of the projects are in the released state. This result indicates that the

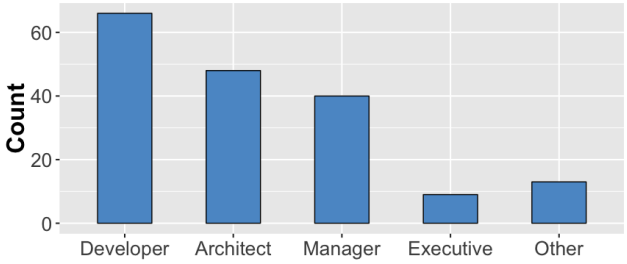

Figure 3: Respondents' Role on Project

majority of the respondents are working on projects that are in phases where software processes are likely to be useful.

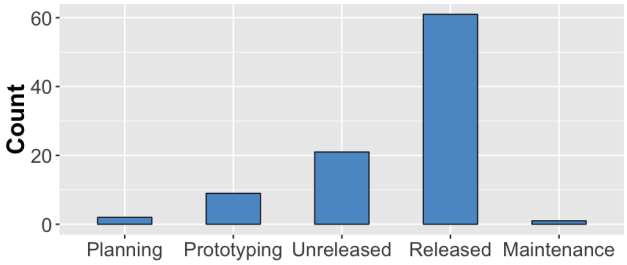

Figure 4: Project Stage

RQ1: To what extent do research software developers follow software process?

To obtain an overall understanding of whether research software developers follow process, the survey asked two questions. First, survey question PQ1 focused on the frequency with which the respondents' teams followed software process. Second, survey question PQ2 focused on the frequency with which the respondents individually followed software process. Figure 5 summarizes and compares the results from these two questions.

Based on the results, we can make the following observations. First, the majority of the respondents indicated that both their teams and themselves individually follow a defined software development process at least *sometimes*, with a large group following process *most of the time* or *always*. It is encouraging to see this relatively high level of process use for both teams and individuals.

Second, examining the overall trend between individuals and teams, we observe that use of process by the respondents' teams and by the respondents individually is consistent at both ends of the spectrum (*never* and *most of the time/always*). The primary differences occur towards the lower middle (*rarely* and *sometimes*, which total about the same when considered as a group). As these scales are ordinal, we performed a Mann-Whitney test to determine whether there is a difference in the responses between these two responses. Not surprisingly based on the figure, the differences are not significant (U = 4403, p = 0.6282).

The lack of a significant difference between teams and individuals allows us to make some general observations. If a team uses process, the individual is likely to use process. If a team does not use process, the individual is not likely to use process. Given that our responses come primarily from smaller teams, this observation suggests that individuals can have an important effect on what a team does when it comes to process (positive or negative). As we did not seek to determine the influence of individual over team (or vice versa), further research may be needed to understand the dynamics of research software teams.

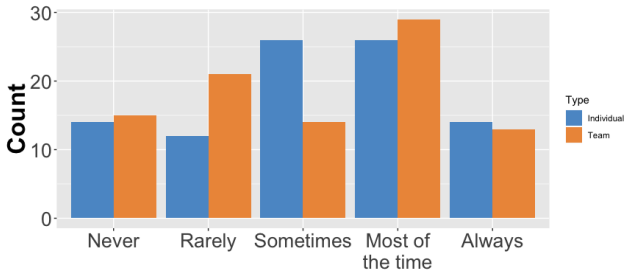

Figure 5: Frequency of following process

Overall, the above findings are encouraging, because if a research team follows a defined process, there is increased potential that team could use process-related metrics as well as other best practices for software development, assuming they collect proper data at all phases of software development.

RQ2: What value do research software developers place on software process?

Survey question PQ3 asks the respondents to indicate the value they see in following a defined process. According to Figure 6, with a few exceptions, most at least see *moderate* value in following a software development process with a relatively large number finding *high* and *very high* value. This result suggests that many research software developers may be interested in following a process for which they see value. The result confirms what we have learned from the literature review, where there are mixed results on which particular software development process research software teams follow, but the common element is that development process is of value to research software teams. While we did not ask respondents to explain why they did or did not values software process, we will explore this question in future studies.

RQ3: Is there any relationship between the valued placed on software process (RQ2) and the use of software process (RQ1)?

It would be reasonable to expect that someone who values software process would be likely to use software process. To check whether this assumption holds, we compared the responses to PQ2 and PQ3. We focused on PQ2 (individually following process) because the respondent has more control over whether he or she follows process than they do over whether the team follows process. In Table 1 we show the distribution of responses between these two variables.

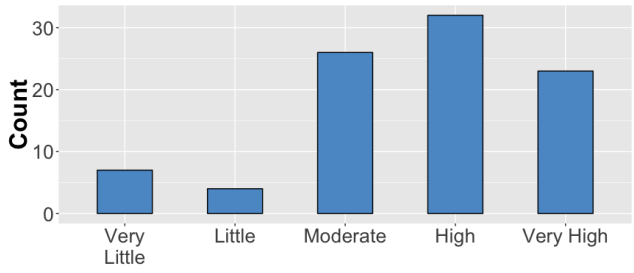

Figure 6: Frequency of value seen in process use

There is a significant (χ^2 (16, N=92) = 92.56, p < .01) relationship between a respondent's perceived value of following a defined software development process and their likelihood of individually following a defined process. This result suggests that one way to increase the level of software process in research software is to help research software developers understand the value of following software process.

Table 1: Perceived value of process vs. Use of process

		Value					
		Very Little	Little	Moderate	High	Very High	Total
Follow	Never	6	1	4	3	0	14
	Rarely	0	1	9	2	0	12
	Sometimes	1	1	11	13	0	26
	Most of the Time	0	1	1	13	11	26
	Always	0	0	1	1	12	14
	Total	7	4	26	32	23	92

RQ4: Which software development processes do research software teams use?

Given the positive perception about following a defined software development process, it is interesting to see which specific process the respondents used. Survey question P4 asked respondents how often the followed a set of popular software process models. The results in Figure 7 show research software developers prefer *Agile* and *Ad Hoc* processes. While many of the respondents used *Use-Case Methodology* and *Rapid Application Development* at least sometimes, most respondents never or rarely used *Waterfall Model* and *Spiral Model*.

It is interesting to note the apparent discrepancy between the relatively high frequency with which respondents follow process (Figure 5) and the relatively low frequency with which the respondents follow any of these process models. It is likely that responds either follow different models or were not aware of the formal names of the processes they follow. In fact, our literature review pointed to the belief that agile-like approaches (not necessarily any of the named Agile approaches) may be a better fit for research software and these results support that belief.

RQ5: To what extent do research software teams use software process metrics?

Survey question PQ5 asked the respondents about the use of process metrics to evaluate the effectiveness of the software development processes in their teams. Consistent with the results reported in our other paper [6] the use of process metrics for evaluation of effectiveness (Figure 8) is quite low. This result is surprising because, without evaluation for effectiveness, teams could be using inappropriate processes. Therefore, one area of potential improvement for research software teams is to employ metrics for evaluating the processes they are using to help determine whether any changes are necessary.

RQ6: To what extent do the demographics of the respondents affect their perception and use of software process?

Using two of the four demographics defined earlier, we analyze whether they impact how respondents' answers about following and valuing process. For each demographic, we split the sample into two categories. We used logic in determining the split points, rather than trying to ensure equal sized groups which were not meaningful. Therefore, because the groups are not of equal size, we normalized the data and analyze the responses as percentages. We describe the analysis separately for each demographic.

Influence of Project Size. We divide the respondents into small teams (less than 5 people) and large teams (5 or more people). From documentation on agile models (e.g. the Scrum Guide [1]) we know that agile teams, ideally, are not so large (between 3 and 9 members). We analyzed the data separately for teams (Figure 9) and individuals (Figure 10). In both cases the differences between large and small teams are significant (Mann-Whitney p < .001).

In general, it is not surprising that large teams followed a defined process more often than small teams (Figure 9). From the literature (and anecdotally) we know that larger teams are more likely to embrace traditional software process.

The reason may be the small number of resources smaller teams get to invest in their software projects. When we look at the data about the effect of team size on whether individuals follow process (Figure 10), the results look a little different. Individuals on small teams appear to follow process more frequently than small teams as a whole.

In terms of value seen in following a defined software development process, Figure 11 shows that large teams tend to see more value in following process than small teams. This result is significant (U = 1951, p < .001). Again, this result makes sense because the larger a team gets, there is more need for a defined process to manage the development.

Influence of Project Stage. We split the respondents based on whether their project was Released (including released and maintenance phases) or Unreleased (Including planning, prototyping, and Unreleased phases).

In terms of following defined processes, the teams of the respondents working on released software followed defined process more than the teams of those working on unreleased software (Figure 12). This difference is significant based on the Mann-Whitney test (U =

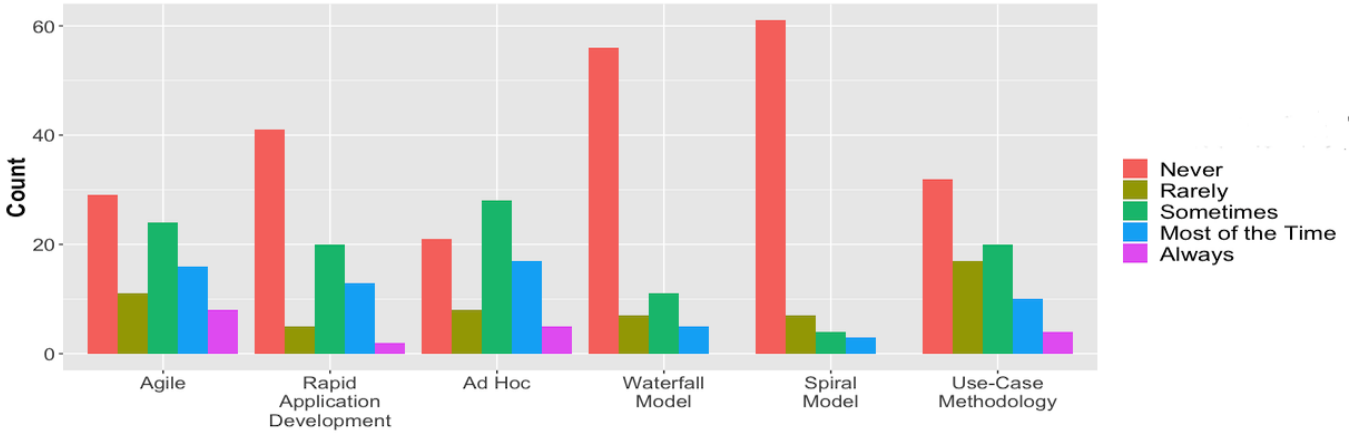

Figure 7: Frequency of using particular process

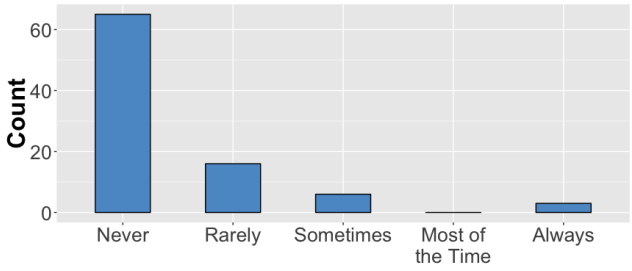

Figure 8: Frequency of process metrics use

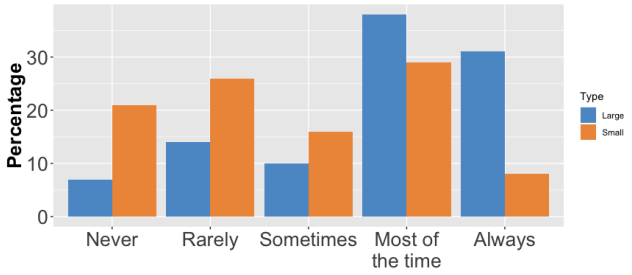

Figure 9: Influence of project size on process following as a team

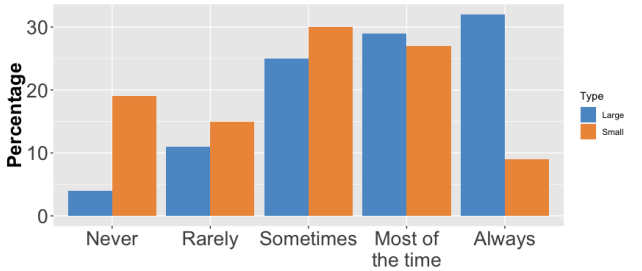

Figure 10: Influence of project size on personally following process

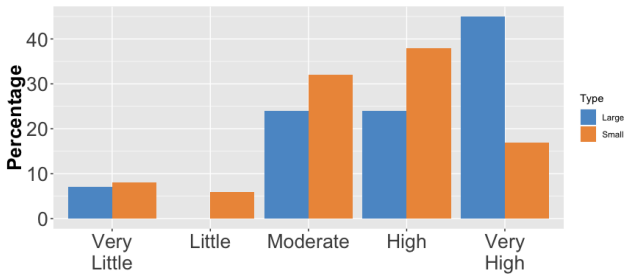

Figure 11: Influence of project size on value seen

5995, $p < .0125$). The results for respondents individually following process mirror the results for the teams.

In terms of valuing process, the results in Figure 13 show that respondents working on released software value process more than those working on unreleased software. The difference is significant based on the Mann-Whitney test ($U = 6135$, $p < .001$). Again, this result is not surprising as those who have released software have had the chance to see the value that following software process can have.

It is also not surprising that people from both the stages tend to not use process metrics to evaluate the effectiveness of the software development process.

5 THREATS TO VALIDITY

Because we collected the data via the same survey instrument used in our previous paper [6], the discussion of validity threats is similar across both studies. But, for completeness, we repeat that discussion in this paper.

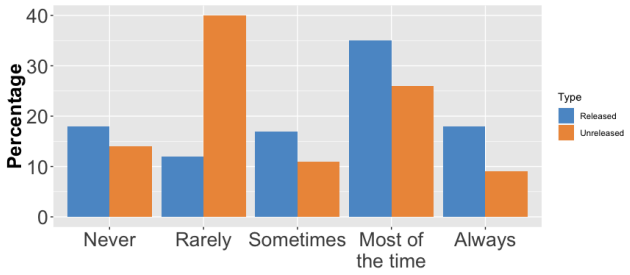

Figure 12: Influence of project stage on process following

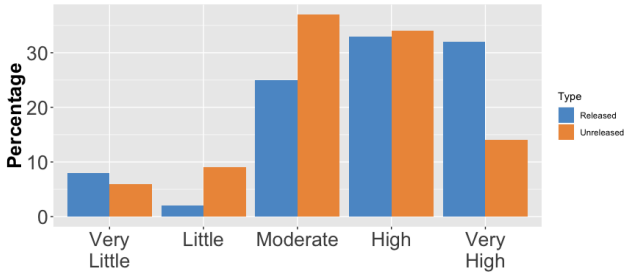

Figure 13: Influence of project stage on value seen

5.1 Internal Threats

There is a potential for introducing bias through the survey design. Because the members of the target survey population are not traditional software developers, it is possible that they lacked the necessary knowledge to properly answer the survey. To prevent introducing bias in this situation, we purposefully phrased survey questions in a neutral manner to allow respondents to answer based on their own understanding.

5.2 External Threats

The survey sample may not be representative of all research software developers. Although we took great care to send our survey to research software developers, due to the particular mailing lists we used, it is possible that some segments of this population, like those from US-based national labs and HPC-related groups are over-represented in the sample. These segments of the population are clearly research software developers, but they may not represent the way all research software developers think.

5.3 Construct Threats

It is always possible that survey respondents misunderstand the survey questions. In our case, however, we went out of our way not only to provide questions but to give clear directions for how to respond to those questions, without biasing the respondents.

6 CONCLUSION

In this paper, we report on the results from 98 respondents to a survey of research software developers to assess knowledge and use of software process. We also report on whether metrics are used to evaluate effectiveness of software process. When it comes to software development process, most respondents reported that they follow a defined software development process and find it valuable. Not only do respondents find software development process to be of value, they actually prefer using a defined software development process. Although the vast majority of responses suggest the use of agile and ad hoc processes, traditional software development processes do see actual use in established research software projects.

An encouraging result is the identified relationship between a respondent's perceived value of using a defined software process and their likelihood of actually using a defined software process. An interesting finding is that the use of agile methods includes the use of lighter weight methodologies, including Use-Case Methodology and Rapid Application Development in their work.

A somewhat discouraging finding is that although respondents see value in process, they rarely use process metrics to evaluate the success of software process. In the absence of process metrics, it is difficult to imagine how quality control can be ensured, because process effectiveness is typically measured at every stage to reduce defects in production. A future area of research is to apply traditional software development processes on actual research software projects and determine the modifications necessary for these projects.

ACKNOWLEDGMENTS

We thank the survey respondents. We acknowledge support from National Science Foundation grants NSF-1445344 and NSF-1445347.

REFERENCES

[1] The scrum guide, version 1. https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf. Accessed: 01-15-19.

[2] J. Carver, D. Heaton, L. Hochstein, and R. Bartlett. Self-perceptions about software engineering: A survey of scientists and engineers. *Computing in Science Engineering*, 15(1):7–11, Jan 2013.

[3] J. C. Carver. Software engineering for science. *Computing in Science Engineering*, 18(2):4–5, Mar 2016.

[4] J. Cates, D. Weinstein, and M. Davis. The center for integrative biomedical computing: advancing biomedical science with open source. In *3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006.*, pages 694–697, April 2006.

[5] A. Dubey, K. Antypas, A. Calder, B. Fryxell, D. Lamb, P. Ricker, L. Reid, K. Riley, R. Rosner, A. Siegel, F. Timmes, N. Vladimirova, and K. Weide. The software development process of flash, a multiphysics simulation code. In *2013 5th International Workshop on Software Engineering for Computational Science and Engineering (SE-CSE)*, pages 1–8, May 2013.

[6] N. U. Eisty, G. K. Thiruvathukal, and J. C. Carver. A survey of software metric use in research software development. In *2018 IEEE 14th International Conference on e-Science (e-Science)*, pages 212–222, Oct 2018.

[7] S. Guatelli, B. Mascialino, L. Moneta, I. Papadopoulos, A. Pfeiffer, M. G. Pia, and M. Piergentili. Experience with software process in physics projects. In *IEEE Symposium Conference Record Nuclear Science 2004.*, volume 4, pages 2100–2103 Vol. 4, Oct 2004.

[8] D. Heaton and J. C. Carver. Claims about the use of software engineering practices in science: A systematic literature review. *Information and Software Technology*, 67:207 – 219, 2015.

[9] E. S. Mesh. Supporting scientific se process improvement. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 923–926, May 2015.

[10] D. E. Post and R. P. Kendall. Software project management and quality engineering practices for complex, coupled multiphysics, massively parallel computational simulations: Lessons learned from asci. *Int. J. High Perform. Comput. Appl.*, 18(4):399–416, Nov. 2004.

[11] M. T. Sletholt, J. Hannay, D. Pfahl, H. C. Benestad, and H. P. Langtangen. A literature review of agile practices and their effects in scientific software development. In *Proceedings of the 4th International Workshop on Software Engineering for Computational Science and Engineering*, SECSE '11, pages 1–9, New York, NY, USA, 2011. ACM.