Understanding Hackathons for Science: Collaboration, Affordances, and Outcomes

Ei Pa Pa Pe-Than(✉) and James D. Herbsleb

Carnegie Mellon University, Pittsburgh, PA 15213, USA
{eipapapt, jdh}@cs.cmu.edu

Abstract. Nowadays, hackathons have become a popular way of bringing people together to engage in brief, intensive collaborative work. Despite being a brief activity, being collocated with team members and focused on a task—*radical collocation*—could improve collaboration of scientific software teams. Using a mixed-methods study of participants who attended two hackathons at Space Telescope Science Institute, we examined how hackathons can facilitate collaboration in scientific software teams which typically involve members from two different disciplines: science and software engineering. We found that hackathons created a focused interruption-free working environment in which team members were able to assess each other's skills, focus together on a single project and leverage opportunities to exchange knowledge with other collocated participants, thereby allowing technical work to advance more efficiently. This study suggests "hacking" as a new and productive form of collaborative work in scientific software production.

Keywords: Hackathons · Time-bounded events · Collaboration · Coordination · Collocation · Scientific software development

1 Introduction

Time-bounded intensive events such as hackathons have rapidly gained traction among both researchers and practitioners in various disciplines due to their potential to leverage collective intelligence, foster innovation, advance technical work, and serve as a ground for future work outside the usual constraints and processes of the workplace [11]. The popularity of hackathons and similar collaborative events has increased dramatically in recent years. For example, collegiate hackathons, just one of the many types of hackathons, in 2017, attracted more than 65,000 participants from 16 different countries[1].

In collocated hackathons, people gather together in the same physical space and form small teams to solve problems within a specified timeframe, typically 2–5 days, leveraging members' diverse backgrounds, familiarity, experience, and expertise. Such events originated in the tech industry where teams consisting of members with a relatively homogeneous background (i.e., software engineers) produced software

[1] Major League Hacking, MLH. https://mlh.io.

N. G. Taylor et al. (Eds.): iConference 2019, LNCS 11420, pp. 27–37, 2019.
https://doi.org/10.1007/978-3-030-15742-5_3

prototypes [7]. Hackathons were subsequently used in other disciplines, such as astronomy[2], biology [13, 17, 19], and polar science [20].

Team collaboration for scientific software production introduces additional coordination and communication challenges due to the involvement of members who are generally trained in two different disciplines: science and computer science or software engineering [1, 6, 16, 19, 20]. However, effective interaction between these two communities is essential not only to foster the correctness and long-term maintainability of scientific software produced [9, 19], but also to cultivate collaboration and the exchange of knowledge between scientists and software engineers.

Previous work on radical collocation [8, 14, 18], which is a situation where team members are together in a room for an extended period of time, suggests that teams collocated in this setting were able to communicate and coordinate their work better, which, in turn, increased productivity and team performance. These studies further noted that radical collocation enabled team members to overhear and participate in conversations, learn from each other, and provide useful inputs, and seek help when needed, resulting in an increase in familiarity among members. Further, face-to-face interaction in teams is known to have a positive influence on participation in follow-up work and facilitate socialization among new members [3].

However, in prior work on radical collocation, teams were either radically collocated for four months [18] or placed in their distant co-workers' workplace for an extended period [8]. These time scales were sufficient for participants to develop familiarity and shared norms that would allow teams to effectively utilize the affordances of radical collocation. This extended radical collocation is quite different from a typical hackathon's 2–5 days. Here, affordances in this study, drawing upon Gibson's concept [4], refer to everything about the hackathon environment that contributes to any kinds of interactions occurred. We examine whether and how hackathons offer the advantages of radical collocation to a team with members of different areas of expertise collaborating on scientific software production. Accordingly, we ask the following two research questions. In the context of scientific software production,

1. How did teams use the affordances of hackathons to collaborate?
2. What were the enduring effects of the hackathons?

To address these questions, using a mixed-methods approach – a combination of interviews and a questionnaire, we studied eight scientific software teams participating in two daylong hackathons held at the Space Telescope Science Institute (STScI)[3] in Baltimore, Maryland. We found that the hackathons offered participants a focused interruption-free workspace, and opportunities for collocated knowledge exchange through which teams were able to concentrate on their work while exchanging technical and/or scientific information, leading the teams to quickly advance their projects. Despite being brief, team members were able to identify others' specific skills and knowledge, thereby possibly enhancing the team overall knowledge about who knows

[2] Astro Hack Week. http://astrohackweek.org/2018/.

[3] The Space Telescope Science Institute (STScI) is operated by the Association of Universities for Research in Astronomy. http://www.stsci.edu/institute/.

what. As participants self-reported that they were willing to adopt hacking practices in their day-to-day work, hackathons could be seen as a new mode of work in scientific software production. In the following sections, we present the background of our study, describe research methods and setting, present our findings, and finally discuss implications of our findings.

2 Background

"Radical collocation" refers to a situation where team members are together in a room for the duration of the project [18]. This strategy was developed in response to communication delays and breakdowns that occurred in distributed software development. Prior studies on software teams have documented the affordances of radical collocation, which – as earlier noted – include: overhearing, spontaneous feedback, learning, ad-hoc collaboration, shared visual space, and increased members' familiarity [8, 18]. These affordances support easier coordination and communication among software team members, resulting in increased productivity and outcome quality.

Coordination is difficult, yet important, for an effective team process [12], and it is even harder when teams consist of members with diverse expertise. Prior studies concerning software work in science have shown that software engineers and scientists tend to approach a task very differently [10, 15, 16]. Software engineers focus on the idea that software engineering methodologies can help assess, test, and improve software correctness [5, 9]. However, scientists are usually not trained or well versed in software engineering practices, and this line of research often concludes with the recommendation that scientists learn and adopt those practices [9]. Scientists, on the other hand, generally just want to "get the plots out," [9, 16], i.e., produce publishable results as quickly and efficiently as possible.

Hackathons are often used as a means to fill the gap between science and software engineering [13, 17, 19, 20]. Prior work on this line of research suggests that hackathons are partially effective in, for example, educating polar scientists how to use high-performance and distributed computing resources, methods, and tools [20], and bringing together software engineers and scientists for skill and knowledge exchange [13]. Since the development of scientific software is a collaborative process of knowledge discovery [15], bringing people from these two different disciplines into the same physical space through hackathons might not only help advance the technical work more quickly but also identify specific practices that work best for scientific software production. However, we do not yet fully understand what kinds of hackathon interactions most effectively encourage collaboration between scientists and software engineers. Therefore, drawing on the theory of radical collocation, this study aims to understand how participants use affordances of hackathons in the production of scientific software, and what are the enduring effects of hackathons in this context.

3 Methods

3.1 The Setting: Two Hackathons at STScI

We studied two hackathons that were held by the STScI in March and May of 2018. The STScI's motivation for running these events was twofold. First, these hackathons were part of the STScI's program of transitioning their tool written in IRAF (Image Reduction and Analysis Facility) scripting language to a Python-based tool due to the discontinued support of IRAF, with the particular aim of assisting this transitioning process. This was also a priority task for the STScI because many instrument scientists and astronomers relied heavily on IRAF for their day-to-day research. Further, scientists were very familiar with the functionality of IRAF, but they were not well versed in Python. Conversely, software engineers were experts in Python but lacked the required domain knowledge. Although both scientists and software engineers were part of the same instrument team in STScI, they all spent most of their day working in silos and their days were fragmented by focusing on various tasks and responsibilities. Accordingly, the second motivation was to bridge the gap between these two groups of people who have different backgrounds (i.e., science and software engineering) by bringing them together in the same physical space for a short period of time to work together on specific projects.

Participants of both hackathons were STScI's employees who were either software engineers or scientists who used IRAF-based tools to perform data calibrations and analyze scientific data, and to provide scientific support to other astronomers dealing with data coming from the instruments installed on the Hubble Space Telescope (HST). The majority of participants knew each other, had worked together before, and had varying levels of Python skills (see Table 1).

Each event was organized by a scientist and held at the STScI's office at the Rotunda in Baltimore, Maryland. Prior to the event, the organizer administered a pre-survey to elicit participants' skills and their project preferences. Based on information it yielded, the organizers assigned participants to specific projects/teams. The organizers then created a shared folder for each team and advised teams to perform preparatory activities such as pre-meetings to identify project goals and tasks, to assign tasks to team members, and to familiarize themselves with the development environment such as GitHub and legacy IRAF code. Prior to the event, the organizers had also sent out tutorial materials related to GitHub, basic Python and Python for astronomy to the teams. On each hackathon day, the participants gathered in the single room, and worked together with team members on a pre-assigned project. Both events, which ran from 9:00 am to 4:00 pm, started with a session during which each team presented the project goals to all participants and ended with a session during which each team reported back to the entire group about what they had accomplished during the event. The two hackathons we studied are described in detail in the following section.

Event A (STIS Hackathon). Event A took place on March 28, 2018. Participants were software engineers and scientists from a team in the STScI that handles data calibration and scientific support activities related to an HST's instrument called the Space Telescope Imaging Spectrograph (STIS). Prior to the hackathon, the event

organizer worked closely with STIS team to identify three projects (A1-3) associated with three highest priority IRAF software packages that needed to be converted to Python. A total of 12 participants took part in this event. They were divided into three teams of four members that worked on one of the three projects mentioned above. Each team contained a mix of software engineers and scientists. All teams met once prior to the event and briefly discussed their projects.

During the event, team A1 which consisted of a software engineer (A14), one scientist with advanced Python skills (A13), and two scientists with basic Python skills (A11 and A12), decided to work in a different room. The group started with a hands-on session about GitHub development workflow led by the software engineer (A14). After this, A13 wrote all the tasks that needed to be implemented in Python on a whiteboard, and divided the labor. A13 performed line-by-line conversion of IRAF to Python while A11 implemented the science-oriented aspect of the module in Python which was then integrated with A13's code. A12 prepared the test data/files for their project. The group worked very closely throughout the day, and A13 and A14 assisted with Python and development-related help when needed.

Team A2 consisted of one software engineer (A24), one software engineer with a science background (A22), and two scientists with basic Python skills (A21 and A23). A21 and A22 started off by identifying the functionality of the existing legacy IRAF code through flowcharts included in their project documentation, and their project consisted of four scripts to be converted to Python. At first, A21 and A22 worked on different chunks of the same script while A22 was occasionally teaching A21 how to write code in object oriented style. The more experienced scientist with basic Python skill (A23) assumed the role of tester and consultant by preparing data/test files and providing science-oriented information when required.

The group process of team A3 was slightly different from those of A1 and A2. A software engineer (A34) introduced the rest of the team to a generalized software package to be used for their project. A software engineer with science background (A32) and a scientist with basic Python skills (A33) tested this package to implement the science-oriented part of their project. A more experienced scientist with basic Python skills (A31) not only provided science knowledge to the team but also helped validate the correctness of the new Python-based program by comparing its results to those produced by his/her own program written in Fortran.

Although teams A2 and A3 were working on different projects, they helped each other by sharing information, especially when they overheard the other group's conversation. Examples of this include referring members of the other team to a specific section in the documentation, and sharing information about specific implementations or calculations that did not work.

Event B (ACS Hackathon). This event took place on May 31, 2018. Participants, again, were both software engineers and scientists who carried out activities related to another HST's instruments called Advanced Camera for Surveys (ACS). One day before the event, the ACS team got together and discussed their hackathon projects. There were six projects/teams in total, of which we managed to interview members

from five projects/teams (B1-5). All projects were "timeboxed", i.e., they were to be completed within a day by the team. A total of 12 participants took part in this event, and each team consisted of three or four members. Each participant in Event B was assigned to at least two projects. In particular, each scientist was appointed as a leader in one project and a consultant in another project. Scientists led all projects except one (B3). All teams worked in the same room most of the day. Each team included members with varying levels of expertise in science and software engineering.

Team B1 was led by a scientist (B11) who brought to the event a pre-written data analysis script in a Jupyter Notebook. He/she advised an experienced scientist who had basic Python skills (B13) to perform end-to-end testing, i.e., including all activities: setting up the environment, running the scripts, and reviewing the explanations and instructions for these scripts. Similarly, a software engineer (B12) also tested the notebook. B11 and B12 troubleshot the errors reported by B13 and opened issues on GitHub, and also provided B13 with help related to both Python and Jupyter Notebook.

The leader of team B2, who was a scientist (B21), worked closely with a software engineer (B22), who also had a science background but little knowledge of Python, but they had not worked together before. B22 worked on optimizing the existing data calibration scripts needed by B21. B21 wrote data analysis scripts and also provided B22 with needed test data/files. Next, B31, a software engineer who had worked with the ACS team before, worked on the project B2 alone while other senior scientists provided him/her with domain expertise. B31 often helped other teams with Python when he/she overheard conversations of other participants. Team B4 was led by a scientist with advanced Python skills, but he/she was not familiar with GitHub (B41). Throughout the day, B41 and B42, a software engineer, adopted a pair programming style in which both worked together to set up a GitHub repository for their project and to convert legacy scripts to Python. The group received Python-related help from B31 and the event organizer (BO1). Another scientist (B51) worked alone on the project B5.

The hackathon model used in event B seemed to encourage both intra- as well as inter-team interaction, observed when participants received help from the other teams' members. Most participants focused only on the projects that they were leading and other projects where they had a consulting role did not reach a state that required their inputs.

3.2 Data Collection and Analysis

The data collection procedure consisted of two phases. In the first phase, we conducted observations during the events where the researcher took detailed notes regarding coordination activities performed by team members throughout the event. In the second phase, immediately after the event, we conducted semi-structured post-event interviews and administered a questionnaire. In post-event interviews, a total of 20 participants (11 from A and nine from B) participated. The topics covered in our interviews include participant's motivations (*"Why did you decide to participate in the hackathon?"*), group dynamics (*"How did your group work together?"*), outcomes (*"How did you perceive the outcome of your project? Did you learn anything at the hackathon that you expect to apply in your daily work?"*), and relationship of hackathon to their day-to-day work (*"How was the hackathon different from how you usually work?"*).

Ten participants completed the questionnaire (four from A and six from B), which elicited participants' satisfaction with the group processes and outcomes on multi-item scales. Participants rated their experience of satisfaction with group processes on five-points scales as inefficient/efficient, uncoordinated/coordinated, unfair/fair, and confusing/easy to understand. Participant's satisfaction with outcomes was evaluated on a five-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree). Example question items include *"I am satisfied with the work competed in my project."* and *"I am satisfied with the quality of my project's outcome."* Two months after the event, we conducted follow-up interviews (post-post-interviews) with one participant from each team. We also interviewed the event organizers to understand the objectives and their expectations of the event. Interviews lasted from 18 to 56 min. Table 1 summarizes the backgrounds of the 22 participants in our sample.

We performed open coding on the interview transcripts and observation notes [2] using Dedoose, a Web-based qualitative data analysis tool. This process resulted in eight codes: ***focused environment***, ***goal-directed workflow***, ***information exchange***, ***team building***, ***meet new people***, ***identify future contact***, ***group process and outcome***, and ***intention to adopt hacking practices in regular workplace***. These codes were the result of a process in which we wrote and shared descriptive memos by interpreting our initial open codes and then repeatedly analyzing them collaboratively to find similarities among the codes in order to develop higher-order codes or categories. We continued this process until no new codes were revealed in our interview data. This process revealed categories related to collaboration practices and the perceived impact of the hackathon on participants and their regular work style.

Table 1. Summary of interview participants (N = 22).

Projects/Teams	Software engineer	Software engineer with science background	Scientist with basic python skills	Scientists with advanced python skills
Event A			AO1	
A1	A14		A11, A12	A13
A2	A24	A22	A21, A23	
A3		A32	A31, A33	
Event B				
B1	B12		B13	B11
B2		B22		B21
B3	B31			
B4	B42			B41, BO1
B5				B51

Note. B1-5 were timeboxed projects. AO1 and BO1 were organizers of Event A and B respectively.

4 Results

Our results revealed how participants made use of affordances offered by hackathons to get their work done, exchange knowledge, extend their social networks and cultivate skills and practices that they could apply in their day-to-day work.

First, participants perceived hackathons as a **focused interruption-free workspace** which enabled them to concentrate on a single project (A32, B20, B11, B21, B41) in contrast to their usual typically fragmented days. Participants also used the hackathon to assess the feasibility of their ideas by attempting to develop them. B20 described how they perceived the hackathon: *"Okay, I'm going to work on this for the entire day, not be interrupted and I'm going to see whether or not this is actually feasible."* (B20). Many participants **felt more directed** in the hackathon as it encouraged them to set and focus on specific goals for the day (A13, A32, B11, B20, B21, B41). A13 noted: *"... it's hard sometimes to find a goal at the end of the day when I come into work that's I need to get to that point. But the hack day is based on that, and at the end of the day you should be at this point, or you should at least try to be at this point."*

Second, hackathons enabled **collocated knowledge exchange** among participants who viewed hackathons as a shortcut to seeking feedback on technical or science-related issues (A12, A14, A21, A32, A33, B20, B31, B41, B42). Otherwise, they would have to formally request assistance from people with relevant skills, who, however, had their own priorities to address and might not be immediately available, causing the progress on work to slow. For example, B42 described how the participants received technical help from others: *"There were a couple times that I had a question about a Python-specific coding thing. One of them was, 'How do I open this file and read the lines out, but I need to read every three lines at a time? How do I effectively do that in a loop?' And [B31], one of the other members of the Hack Day, helped"* (A31). Similarly, A21 noted: *"... being in the same room made the interaction much faster in that someone could ask a question right away. Or – and as soon [A23] had some code written, I could test it. So in that sense, it was certainly much faster turnaround and more efficient in that sense because everyone's mind is fresh."* (A21). Similarly, one scientist (B41) noted how working side-by-side with a subject matter expert was exceedingly helpful and productive: *"I think that I normally wouldn't have had that many hours of [B42]'s attention on any given day and [even though] I could've done that coding myself, ... it would've taken a lot longer."* (B41).

Third, several participants noted how hackathons enhanced **a sense of team identity** and described their experience as "some good synergy that develops it as a team." (A22). Other members commented that it is typical to *"feel united when you're all working together on one issue in a little bit more of a relaxed environment"* (B21), and noted *"the sense of community and a sense of we're all working towards this common goal of getting stuff done for the ACS team"* (B42). Despite being a brief collaboration, participants seemed to be able to **identify go-to persons for future**—persons with required skills who they could ask to help them with certain issues and problems. This was especially the case when the hackathon teams involved members who were outside of their regular workgroup (A30, A40). For example, A40 noted: *"I think it [the hack day] just sort of gave me a little bit more confidence to go talk to – who to approach"* (A40).

Further, brief interaction with active contributors of open communication channels (e.g., institution-wise python channel) during the hackathon seemed to have reduced a perceived barrier for scientists to take part in those channels (B41).

Fourth, many participants commented on having encountered several **learning opportunities** by working side-by-side with subject matter experts during the hackathon (A13, A14, A31, B21, B22, B41). The team members also overheard conversations with other teams and provided **useful technical and/or scientific information**. A31 recalled the participant's experience as follows: *"We were talking about a particular issue. One of the things that goes on in this code, and somebody from the other group kind of chimed in from his experience, giving some additional perspective on that particular issue."* (A31). Likewise, scientists who knew Python but were not familiar with software engineering learned **best practices and development workflow** (e.g., object oriented style coding and GitHub (A11, A23, B13)), as well as **tool-specific nuances** (e.g., using multiple languages in a Jupyter Notebook (B41)). In some cases, the actual learning process did not take place during the event, but subsequently. For example, participants (B11, B12) reported that they aimed, in the future, to explore more about tools that they learned about from other participants (e.g., Ginga – a Python toolkit for building viewers for scientific image data).

Finally, several participants indicated that participating in the hackathon made them realize how they could be more productive during their regular workday, and that they **intended to use hacking practices in their regular work** (A13, A22, A23). Participants of both events appreciated the value of the hackathon and suggested having more hackathons with the entire team at least twice a year. Some participants reported that they had discussed the possibility of using common free time to work side-by-side like a mini-hackathon (A22, A23). Further, the post-post interview conducted with A11 (the team lead of STIS) revealed that the team had organized another hackathon and planned to organize one every two months. A participant of Event B (B11) had been organizing "lunch hacks" every Thursday, where attendees brought in issues to solve together with other attendees. One participant (A22) expressed how he imagined hackathons as the future of work: *"I think that it was very useful in helping us see how productive it can be. And I think, that's only going to make us feel like it's a way we could – it's something we could bring to the way we work in the future"* (A22).

Questionnaire Results. While the small number of participants does not support meaningful statistical analysis, we present our questionnaire results to enrich our qualitative descriptions. The analysis of questionnaire data revealed differences in satisfaction between participants of these events. Participants of Event A ($M = 3.88$; $SD = 0.66$) were slightly more satisfied with their team process than those of Event B ($M = 3.67$; $SD = 0.58$). In contrast, participants of Event B reported having higher levels of satisfaction ($M = 4.47$; $SD = 0.45$) with the outcome than those of Event A did ($M = 3.67$, $SD = 0.58$). This finding corroborated our interview data in which several participants of Event B (B21, B42, B51) expressed the view that being able to get the work done felt fulfilling and rewarding, whereas a participant of Event A expressed a little disappointment noting that the *"project was not the one that could be completed within a day"* (A21).

5 Discussion and Conclusion

We found that brief intensive collocated collaboration or hackathons can effectively bring two communities with different expertise for productive interactions on specific issues and problems, resulting in quick progress on technical work or projects. Participants perceived an increase in productivity that they attributed to being able to work in a focused interruption-free space that enabled them to concentrate on one thing at a time, and then discuss technical and/or science-related matters with other participants. This finding is consistent with prior work on radically collocated teams for extended periods of time, which demonstrated that affordances of radical collocation enabled easier coordination, and reinforced relationships among members which, in turn, increased team performance [8, 14, 18].

In addition to being a space for "getting the work done", hackathons facilitated opportunities for knowledge exchange and serendipitous learning, which often happened by overhearing conversations among other participants. Learning also was seen to happen when a scientist with little software engineering experience was paired up with a software engineer or a scientist with advanced Python skills. In other words, our findings suggest that hackathons could be used as an integral element in the process of producing a consistent set of scientific tools for long-term maintainability. This process requires an intensive collaboration between scientists and software engineers [9, 19]. Here, hackathons have an ability to bring in scientists, who typically work in silos either focusing on their own scientific research agenda or having insufficient time to work on needed tasks due to various responsibilities, to the same space with software engineers in order to collaboratively discover scientific knowledge while following and adjusting best software engineering practices for their scientific needs.

Despite their being brief, hackathons enabled participants to explore each other's skills and expertise and identify future useful contacts, perhaps suggesting that hackathons can enhance the team's transactive memory [21]. Taken together, our results suggest that in order to advance technical work more quickly and effectively for scientific software teams embedded in an organization, it is helpful to give them common free time, and that doing so is beneficial not only for getting the work done but also for enabling knowledge exchange and learning among members.

As a general rule, we found that it is important for participant satisfaction to have the scope of the project aligned with the event time frame and team size (i.e., time-boxed project). As reflected in the questionnaire results, teams that performed time-boxed projects were more likely to be satisfied with their project outcomes. On the other hand, participants might be motivated to return to, and finish the projects that they were not able to complete during the event. This calls for future research as we were not able to draw any conclusions based on our current data. We found that the teams we studied have continued running weekly mini-hackathons, and that participants appreciated a way of solving problem on the fly. The experience as a whole seemed to have added "hacking" as a new way of working, a new element in the team "toolbox" they could use as the need was perceived. Nonetheless, future research is needed to investigate whether hackathons introduce other changes in the way that people work.

References

1. Bos, N., et al.: From shared databases to communities of practice: a taxonomy of collaboratories. J. Comput.-Mediat. Commun. **12**, 652–672 (2007)
2. Corbin, J., Strauss, A.: Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory, 4th edn. SAGE Publications Inc., Thousand Oaks (2014)
3. Crowston, K., Howison, J., Masango, C., Eseryel, Y.: Face-to-face interactions in self-organizing distributed teams. In: Presentation at the OCIS Division, Academy of Management Conference (2005). http://floss.syr.edu/StudyP/ftf2005.pdf
4. Greeno, J.G.: Gibson's affordances. Psychol. Rev. **101**(2), 336–342 (1994)
5. Hatton, L., Roberts, A.: How accurate is scientific software? IEEE Trans. Software Eng. **20**, 785–797 (1994)
6. Heaton, D., Carver, J.C.: Claims about the use of software engineering practices in science: a systematic literature review. Inf. Softw. Technol. **67**, 207–219 (2015). Carver
7. Henderson, S.: Getting the most out of hackathons for social good. In: Rosenthal, R.J. (ed.) Volunteer Engagement 2.0: Ideas and Insights Changing the World, pp. 182–194 (2015)
8. Hinds, P.J., Cramton, C.D.: Situated coworker familiarity: how site visits transform relationships among distributed workers. Organ. Sci. **25**, 794–814 (2013)
9. Howison, J., Herbsleb, J.D.: Scientific software production: incentives and collaboration. In: The ACM 2011 Conference on Computer Supported Cooperative Work, pp. 513–522. ACM, New York (2011)
10. Kelly, D.: Scientific software development viewed as knowledge acquisition: towards understanding the development of risk-averse scientific software. J. Syst. Softw. **109**, 50–61 (2015)
11. Komssi, M., Pichlis, D., Raatikainen, M., Kindström, K., Järvinen, J.: What are hackathons for? IEEE Softw. **32**, 60–67 (2015)
12. Kraut, R.E., Streeter, L.A.: Coordination in large scale software development. Commun. ACM **38**, 69–81 (1995)
13. Möller, S., et al.: Community-driven development for computational biology at sprints, hackathons and codefests. BMC Bioinf. **15**, S7 (2014)
14. Olson, G.M., Olson, J.S.: Distance matters. Hum.-Comput. Interact. **15**, 139–178 (2000)
15. Paine, D., Lee, C.P.: Who has plots? Contextualizing scientific software, practice, and visualizations. ACM Hum. Comput. Interact. **1**, 21 (2017). Article 85
16. Segal, J.: Scientists and software engineers: a tale of two cultures. In: The 20th Annual Meeting of the Psychology of Programming Interest Group, PPIG 2008, University of Lancaster, UK (2008)
17. Stoltzfus, A., et al.: Community and code: nine lessons from nine NESCent Hackathons. F1000Research **6** (2017)
18. Teasley, S., Covi, L., Krishnan, M.S., Olson, J.S.: How does radical collocation help a team succeed? In: The 2000 ACM Conference on Computer Supported Cooperative Work, pp. 339–346, ACM, New York (2000)
19. Trainer, E.H., Kalyanasundaram, A., Chaihirunkarn, C., Herbsleb, J.D.: How to hackathon: socio-technical tradeoffs in brief, intensive collocation. In: The 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, pp. 1118–1130. ACM, New York (2016)
20. Wyngaard, J., Lynch, H., Nabrzyski, J., Pope, A., Jha, S.: Hacking at the divide between polar science and HPC: using hackathons as training tools. In: 2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 352–359. IEEE (2017)
21. Zhang, Z.-X., Hempel, P.S., Han, Y.-L., Tjosvold, D.: Transactive memory system links work team characteristics and performance. J. Appl. Psychol. **92**, 1722 (2017)