Hackathons in software engineering education – lessons learned from a decade of events

Jari Porras
Lappeenranta University of Technology, Finland
jari.porras@lut.fi

Jayden Khakurel
Lappeenranta University of Technology, Finland
jayden.khakurel@lut.fi

Jouni Ikonen
Lappeenranta University of Technology, Finland
jouni.ikonen@lut.fi

Ari Happonen
Lappeenranta University of Technology, Finland
ari.happonen@lut.fi

Antti Knutas
Lero, the Irish Software Research Centre, Glasnevin, Dublin, Ireland
antti.knutas@dcu.ie

Antti Herala
Lappeenranta University of Technology, Finland
antti.herala@lut.fi

Olaf Drögehorn
Hochschule Harz,
Germany
odroegehorn@hs-harz.de

ABSTRACT

Hackathons are currently a hot topic in industrial learning settings. Like intensive collaborative courses (e.g. code camps), hackathons have been shown to be successful tools for learning. However, current research has failed to adequately compare the two approaches with respect to who benefits, which stakeholders are involved, and what the practical arrangement differences are. We used a literature review, our own multi-year learning experiences, and written and interview material from students and industry participants to present an overview of hackathons and code camps. Based on the results of our analysis, we present a taxonomy, based on our experiences, to help practitioners decide which kind of intensive event approach is suitable for them, depending on their industry and educational needs. This synthesis and the study results provide the first steps towards a functional definition that covers intensive collaborative working events involving real-life problems, such as code camps, hackathons, and 24-hour innovation workshops. Currently, the terminology is diverse, but there are commonalities and differences across each of these events and their purposes.

CCS CONCEPTS

• Applied computing → Education Interactive → Collaborative learning; **• Applied computing → Education Interactive → Interactive learning environments**

KEYWORDS

Hackathon, Code camp, outcomes, industry collaboration, stakeholder perspectives, learning, software engineering

SEEM'18, May 27-June 3, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5750-0/18/05…$15.00
https://doi.org/10.1145/3194779.3194783

J. Porras, J. Khakurel, J. Ikonen, A. Happonen, A. Knutas, A. Herala, and O. Drögehorn. 2018. SIG Proceedings Paper in word Format. In Proceedings of ACM ICSE conference, Gothenburg, Sweden, May 27 - June 3 2018, 8 pages. DOI: 10.1145/3194779.3194783

1 INTRODUCTION

University education and software engineering education are rapidly evolving, and requirements for both hard and soft skills are changing. On the hard skill side, Association for computer Machinery (ACM)/ Institute of Electrical and Electronics Engineers (IEEE) curriculum guidelines (both on CS and SWE) aim to capture the changes behind the trends, such as the increasingly interdisciplinary nature of software engineering [34] and the growing role of big data and computational sciences in computer science curricula [33]. On the soft skill side, the emphasis is shifting toward the role of teamwork, end-user awareness, and the context of the software solution [34]. Universities and industry do not always see the importance of different skills in the same way [14]. Learning itself is also becoming more interactive, and the importance of collaborative learning and teamwork is growing [25]. At the same time, intensive courses and team-based rapid development methods are growing in popularity within software engineering education. These approaches teach students to cooperate in groups and to help one another achieve their learning goals by collaborating (e.g. by sharing and applying acquired knowledge to improve group work). These methods have been proven to work in tertiary education in both domestic and international studies [9, 27, 28]. Furthermore, collaboration itself has become an important subject in education [15] and an essential 21st century skill [2, 6]. In a collaborative learning environment, students can experience new approaches to thinking by listening to their peers and obtain clearer perspectives of different topics by expressing their understanding [12].

These project-based approaches are often connected to real-world problems or challenges proposed by industry representatives

[18]. Various problem- or project-based learning (PBL) courses [11] and capstones [26] combine the above-mentioned aspects to help students achieve desired learning outcomes. Team-based approaches help students understand the value of a wide skill set, different personalities, and diverse study/empirical knowledge backgrounds [29]. A Belbin team analysis [3] can be used to gain more detailed information on how such teams are structured. However, while understanding a team's structure may help explain the team's outcomes, it might not affect the actual teamwork.

The collaborative learning events come in different sizes, shapes and names. In general, we call these events hackathons but will show the difference behind different names. The aim of this study is to explore the various approaches to implementing hackathons and their outcomes for different stakeholders in the context of software engineering education and to present taxonomy, based on our experiences, for the various types of collaborative learning events. This study is based on real experiments and observations from hackathon events from the early 2000s to the present. We aim to share our lessons learned on the following issues: 1.*At what level can hackathons be used in software engineering education to teach students necessary skills and competencies?;* 2.*What kinds of outcomes do different types of hackathons provide for stakeholders?*

We aim to show our experiences on these issues by analyzing several hackathon-type events we have arranged or participated in since 2003. Today, these intensive collaboration-oriented events or competitions are often called hackathons or hacks, but they can also be called by other names, such as code camps, innovation camps, dev days, boot camps, etc. These names represent a variety of meanings. Most of the terms originated in the late 1990s or early 2000s. In this paper, we use the various names as they were used in the original events; however, all can be grouped under the common term "hackathon." While studying the learning outcomes of these different types of events, we also created a trajectory of how the events have evolved over time and whether this evolution affects how hackathons can be used in software engineering education. Based on the results of these experiments, we will present learned lessons and give recommendations for hackathon utilization.

The rest of this paper is structured as follows. In the next section, we review related work as presented in recent literature. Then, we present a taxonomy of code camps ("educational hackathons") and discuss lessons learned. Finally, we offer conclusions.

2 LEARNING THROUGH INTENSIVE AND COLLABORATIVE PROBEM-SOLVING EVENTS-HACKATHON

This section details the collaborative learning, different structures, and goals of the hackathons discussed in recent years by other researchers. Collaborative learning is an educational approach in which students at various performance levels share ideas, solve problems through critical thinking, and work in small groups toward a common goal (i.e., completing a given task or developing a product [13]). Collaborative learning has been stated to enhance drill-and-practice skills and critical-thinking skills [13]. Laal et a. [22] categorized the benefits of collaborative learning into four dimensions: social, psychological, academic, and assessment-based. With the consideration of those benefits, teaching professionals are holding collaborative social learning events such as 'hackathons', 'hack days', or 'coding' or 'dev camps' to foster twenty-first-century learning skills, which aim to create usable products or software by fostering team-based creativity and innovation [8].

A case study presented by Ward et al. [35] used a student code camp as a bottom-up design approach to generating mobile library apps. The authors found that it was beneficial for both the library and the students. For example, the library felt that the camp was useful to their ongoing mobile app development process in terms of the solutions created and also the feedback gathered on what types of apps students wanted in order to improve their future user experiences. Similarly, the students found that they had an opportunity to meet people with similar coding interests, and they acquired new knowledge and skills (i.e., specific coding languages, user interface design, team building, etc.—areas in which they had little experience). Kilamo et al. [19] described a two-week code camp in which the students used a collaborative online-integrated development environment to create different kinds of web services. They found that the students benefited from the collaboration between the teams which allowed them to gain a wider knowledge—and the balanced workload. They stated, "The code camp acts as a community of practice where learners learn not only from the direct software engineering activities but also through sharing information and experiences among themselves." This is also supported by related research on communication patterns among students in collaborative software engineering courses [20]. Another study examined the informal learning aspects of hackathon events and showed that these events help computer science and engineering students to compress their development process into a short time frame [24], allowing them to hone their problem-solving, project-management, and prioritization skills while still having a fun, enthusiastic, and engaging experience. In addition, one event helped industry participants to recruit students, generate long-term projects, and engage participants from other companies. The event also helped a university to hire research assistants to develop apps and collaborate with the industry on future projects. One study [10] presented the outcomes from Think Global Hack Local (TGHL) which is a non-competitive, community based hackathon that involves non-profit organizations and student developers. They found it was beneficial for both students as well as non-profit organizations. E.g., the students did feel it as an opportunity i) to have collaboration with the community; ii) to meet new people with similar interests; iii) to learn from one another; iv) to work on problems with a goal to solve real-world problems. Similarly, non-profit organizations felt that it was beneficial to get connected with students to support their learning and also to create and visualize something that has only existed as a concept [10]. Also, dev camps can be used for project-project based learning scenarios, as shown in as study where dev camps where co-organized by a company and an university [8]. The facilitator accessed the outcome of the projects on a five-point Likert scale and found that all the results

went beyond expectations and fulfilled six dimensions of quality: creativity, novelty, motivation, plausibility, technical soundness, and understandability. They point out that such events are beneficial (i) for students, who can acquire new knowledge and skills (i.e., those who possess limited expertise or technological skills can learn from skillful group mates, and high-performing students can practice by guiding their teams); (ii) as a bridge between education and industry. However, the researchers also pointed out that a negative aspect of such an approach is that the learning process cannot be predefined, and skills cannot be standardized.

2.1 Sample Fabrication

The hackathons are controlled events that usually focus on software development and the term was coined by the open source software developers in 1999 [5] and combines the terms "hacking" and "marathon" [21]. Because of the short age of the term and use of many related terms, there are multiple definitions about the events and their structure, even the tools and methods used [30]. The key point, where the hackathon organizers do agree on, is that the fundamental goal of a hackathon is to engage open innovation and allow external and interdisciplinary experts work with one another to create something new [7]. Hackathons can be seen as a straightforward tool to engage open innovation and collaboration while acquiring feedback about the issues that require solutions. Further, [5], state, "The greatest potential and value of hackathons is in providing an opportunity for people to meet and collaborate to create new links in the medium to long term, rather than the short term focus of the event" (pp.11). All forms of hackathons involve a variety of stakeholders, ranging from educational institutions, research organizations, government departments and cities to cultural institutions and other non-profits [5].

Hackathons can be divided into multiple categories while preserving the structure and overall theme of hackathons. Literature recognizes hackathons with different titles based on the goals, such as civic hackathons [17], industrial hackathons [30], educational hackathons [24], TGHL [10] and even cultural hackathons [4]. Out of these types of hackathons we are mostly interested in education and industrial types due to our experiences during the last decade and because we emphasize the university industry collaboration. For example, education hackathons engage at least universities and students and possibly others, such as industry representatives, to cultivate long-term relationships as industry–university relationships and to share in-depth technical or business knowledge. Similarly, TGHL engages non-profit organizations and student developers to solve problems that organizations would otherwise lack the resources to solve [10].

3 A DECADE OF EXPERIENCE ARRANGING HACKATHONS

Hackathons come in many forms (e.g., tech-centric or focus-centric [5]), sizes, and names, but all seek to produce a minimal viable product (MVP) by the end of the event [31]. As the following subsections show, we have piloted and run hackathons for a variety of purposes. This section aims to describe the various implementations of hackathons to offer ideas on how hackathons can be used. We use the original names of the events and categorize all of them under the term hackathon. The lessons learned from these cases are discussed in the next section.

3.1 Code camps arranged by LUT – focus on learning technologies

Lappeenranta University of Technology (LUT) started using code camps as a teaching method as early as the summer of 2003. In the beginning, LUT's code camps were arranged as 24-hour continuous working events that took place at the end of the annual summer school on telecommunications. Since the summer school emphasized different themes (e.g., the newest communication technologies) each year, the code camps were designed to address challenges and opportunities relating to the technologies surrounding the selected theme(s). The first ever LUT code camp focused on Symbian programming and was arranged together with Digia Ltd., a major subcontractor of Nokia at the time. The aim of these 24-hour code camps was to collaboratively learn a given technology while simultaneously working on a predefined challenge and producing solutions in a given topic area. Although each code camp selected a winning team, just as current hackathons do, the educational emphasis was on collaborative teamwork and learning. Therefore, the evaluation process emphasized having a code camp spirit and being able to work with others, giving the sophistication of a given solution only half of the evaluation points. The LUT code camps were successful in teaching students valuable hard skills (e.g. knowledge on new technologies) and soft skills (e.g. teamwork). Furthermore, although the 24-hour code camps worked mainly as short training events, they had benefits for both i) industry participants (i.e. by introducing them to students with scarce and newly developed skills) and ii) students (i.e. by increasing their knowledge on technological trends, job market requirements, and soft skills like teamwork, adaptability, task allocation, and critical thinking). For example, one student wrote that having an "*[i]ndustry expert as lecturer for a day was huge plus (more efficient way to learn real life practices than with academic lecturer).*"

However, the main challenge of a 24-hour event was that students had very limited time to interact with industry experts or learn valuable hard and soft skills relevant to software engineering, such as teamwork, adaptability, task allocation, critical thinking, and knowledge about the actual technology. Furthermore, the final prototypes often worked so well that there was no need for new innovative output, meaning that the companies had less opportunity to test the students' abilities or engage deeply with the students to generate new ideas and iterate innovative solutions. For example, even when industry partners agreed during initial discussions that a one- to two-day code camp would be the "right length" for a task, they later said that the camps did not generate "surprising new innovations." To address such feedback, LUT started experimenting with one-week and five-day code camps. The first week-long code camp was arranged with Nokia in the spring of

2005 (just after experiences on the implementation of two 24-hour summer school code camps). The main idea of the code camp remained the same, but more effort was directed toward improving the students' hard and soft skills and giving the students new learning and coding practice by allocating time for interactions among students, Nokia participants, and teachers. For example, whereas, in 24-hour code camps, a demonstration of the use of a technology or platform was enough, in week-long code camps, solutions had to be far more technical, better presented, and generally elegant. Students received and completed pre-tasks and reading, and the code camps were more like intensive real-life practical exercises with highly emphasized collaboration elements and software solution delivery deadlines. Students also had more time to interact with industry participants during social events and within the classes themselves. The typical structure of a week-long code camp is presented in Figure 1.

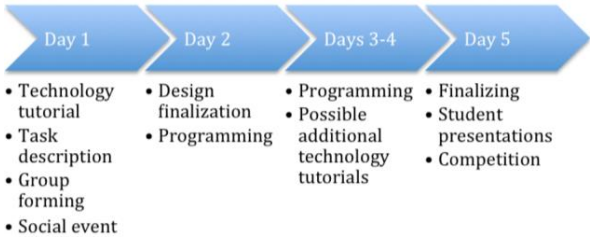

Figure 1: Typical structure of a LUT Code Camp.

An analysis of data collected from the first code camp shows that 76% [27] of the students responded positively to code camp-style events. For example, one student who participated in a code camp reflected that "*[r]elevant 'best practice' tips with instant action testing in start of the Code Camp connect the theory directly into the practice, and instant continuation to the own project made learning easier.*"

In response to students' positive reactions, LUT has continued to implement code camps as a viable teaching method for both hard and soft skills. Since the university's goal is to impart basic knowledge, skills, and competencies on software engineering, code camps offer useful opportunities to introduce rising technologies and platforms (e.g. .NET, Maemo, Meego, Qt, AR, ReactJS, etc.) in collaboration with interested companies (e.g. Nokia, IBM, Microsoft, SUN, Google, CGI, etc.).

In addition, during the last three years, LUT has started to implement 24-hour code camps for new/freshmen students to help them get to know one another and build team spirit (a new type of learning objective) during the first week of their studies. This approach encourages students to embrace the attributes needed for successful teamwork [32], such as i) commitment to team success and shared goals, ii) interdependence, iii) interpersonal skills, iv) open communication and positive feedback, v) appropriate team composition and commitment to team processes, vi) leadership and accountability, and vii) task synchronization.

To accomplish these purposes, LUT uses the platform "Robocode" (http://robocode.sourceforge.net/) to program virtual war/fighting tanks with one-to-one and one-to-all battles. The platform is easy to use and supports several different strategies and methods to achieve combat success. Within this setting, prior to their final battle, teams explain their strategies and designed battle tactics for the other participating teams. Although the final battle results are statistically analyzed to determine the winning team, the collaboration within and among teams fosters team spirit among all freshmen (soft skills) and helps students get a good start on their studies. The competitive format of the event produces a "push" (i.e. put more effort into studying) spirit, and the code camp atmosphere provides a "safe" environment in which students can compete with soon-to-be classmates while building friendships (rather than losing them).

3.2 Code camps arranged at Harz - changing the objectives of the Hackathon

Harz University of Applied Sciences started to arrange code camps around 2012 for two different purposes, each distinct from the ones pursued by LUT. Both approaches show how flexible code camps can be for suiting different purposes when the learning outcomes are properly planned.

Harz implements 48-hour code camps that focus on increasing students' familiarity with various technologies and topics (also called specialization code camps). The code camp facilities are filled with various technological solutions, ranging from home automation equipment to programmable Legos. Both participating students and interested citizens can play/work together on various technologies. Together, the students and citizens i) influence the structure and organization of the society in which we live [1], ii) make teams more dynamic, and iii) network with one another. The learning outcomes of this approach include knowledge of multiple technologies, an improved understanding of societal problems, and a clearer picture among students of what their interests are and what they should select as their specializations. Thus, these code camps differ from the previously mentioned approaches.

Harz also uses code camps as a means for students to show off their competencies. Computer science and, especially, software engineering skills are better tested through actual work than exams. Exam tests (usually) focus more on memory, expression, and writing skills than the skills needed in working life (e.g. providing/engineering a solution for a specific task, given the freedom to choose how to move from the input to the needed output without strict limits on creativity in implementation). Therefore, a code camp "exam" approach is a meaningful way of testing students' knowledge, skills, creativity, and competencies.

3.3 Code Camp Competitions – a step towards product development

Code camps have been successfully used as gateways to various competitions (i.e. as tools for building students' competence to enter such competitions victoriously). These code camps are arranged similarly to the week-long code camps described above, but their objectives go beyond the actual code camp course. In short, code camps can be built around competitions, such that the code camps resemble industry hack events or more traditional types

of learning events, but with a twist: Students are simultaneously learning how to compete in an open challenge event. For example, from 2007 to 2012, LUT combined its code camps with Microsoft's international Imagine Cup competition. The infusion of code camps into open challenge events was i) an excellent preparation for the competition and a fulfillment of educational outcomes; ii) a driver for students to learn new hard skills, such as programming languages (e.g. .NET and other related technologies), and soft skills (e.g. creativity, team work, and adaptability) to compete with local semifinalists and international competitors; iii) an opportunity for students to enjoy new experiences; and iv) real product development. Although such events had benefits, they also suffered several challenges. For example, the process of exploring problems together with stakeholders used several soft skills, such as interpersonal skills. Similar approaches have recently been used in NASA's space app competition, the Green Code Lab Challenge, and many more. Competition-based code camps emphasise real industry and society needs and, as such, pave the way from educational hackathons to more commercial industry hacks.

3.4 Industry hacks – focus from universities to industry

Industrial hackathons began to evolve in the early 2010s. With the open data movement, companies saw possibilities to "crowdsource" new services and solutions. Many companies had previously worked with universities in educational hackathons, and they evolved these experiences into more industry-oriented hackathons. The focus of the events shifted from educational fun to more business-oriented activities. In other words, though students were still invited, the focus was on business rather than education. Teams were assumed to already know the technologies used, and no educational guidance was given. These events were also less collaborative, especially when they involved multiple small startup companies all trying to prove their skills and get their solutions selected and implemented. Because of this format, smaller companies saw industry hackathons as an opportunity to network with bigger players, and bigger companies saw them as a way of shopping ideas. When an industry hack is organized as a serious business development event by an outside company that is compensated for its work, it is designed not to recruit students, but, rather, to shop for ready-made solutions (e.g. from start-up firms). Such events also tend to be more serious and less fun than educational hackathons. There are, of course, exceptions to this phenomenon. The industry hacks have also shaped our perception what hackathons mean and how we use the term. In the biggest events, for example, large companies organize such challenging tasks that it is usually not possible for any one startup to offer a standalone wide-scope solution. Such events, therefore, foster more collaboration and shared learning.

4 TAXONOMY OF OUR HACKATHONS

Hackathons, and especially educational hackathons, can be seen as playing two roles: improving education and serving as teaching tools. Hackathons can be used to innovate and develop new teaching methods. They also support a larger set of stakeholders and participants than regular educational meetings [5]. Hackathons as teaching tools—referred to in most of our cases as code camps—offer students new ways of learning specific technologies or methods in short periods of time through practice [5]. Each hackathon has different goals, which affect the name used. In educational hacks, the goal is to learn, rather than to create a fully functional solution to a problem. However, it is not forbidden to do both. Since the goals of such events tend to lean towards learning, it is only natural that the main body of participants is usually students. In some cases, these students are adults with years of industry experience. Here, the nature of participation changes, and the solutions tend to be more finalized (i.e. the scope of details included in the solution tends to be wider). In the Table 1 we summarize the various code camps and their aims, stakeholders, and outcomes.

From a cost point of view, it can be generalized that learning focused and academic style code camps (e.g. Freshmen and Exam camps) are usually "cheap" or free for all participants. Of course, participants must invest time into participation, but there are generally no participation fees or other arrangement costs for students and/or companies. On the other hand, industry hacks are organized by commercial companies with full-time employees and salaries to pay. Usually, the party that covers the costs is the one to receive the results (i.e. typically the corporation or large company that promotes the event to startup companies).

Table 1: Taxonomy of our use case hackathons

Hackathon	Aims	Stakeholders	Outcomes
24-hour	Fast intro to new topics (e.g. those not available in current curricula). Learning by doing.	Universities, students, companies	Implementation, improved skills, new knowledge, fast prototypes
Week-long	Understanding the workings of selected topic. Learning by doing.	Universities, students, companies	Deep knowledge and skills of the event theme, working prototypes
Team building (Freshmen)	Bonding and acquaintance. A good start on studies.	Universities, students	Understanding of own strengths and weaknesses in programming.
Specialization	Getting a glimpse of multiple different technologies.	Universities, students, citizens	Technology and operability knowledge
Hack as an exam	Testing skills in a real project environment.	Universities, students	Knowledge of the skills each student has

Competition	Innovating solutions to a given challenge and first implementation. Emphasizing innovativeness.
Industry hack	Working and providing solution to a company-defined theme.

5 ROLE OF HACKATHONS IN SOFTWARE ENGINEERING EDUCATION AND LESSONS LEARNED

Both IEEE and ACM curricula for software engineering and computer science [33, 34] emphasize practical implementation-focused and group-based courses (e.g. capstones or hackathons) that give students opportunities to utilize their knowledge in "real" projects. Although even when hackathons do not span multiple months (like capstone courses) they do have the key characteristics: a reflection on knowledge, teamwork, an implementation orientation, and an outcome evaluation. Hackathons have external "customers" (in the example cases, from either industry or society) that make them more real for the students. Thus, hackathons have the potential to fulfill most of the learning / educational objectives typically set for capstones.

In the following, we summarize the lessons learned and explore both positive outcomes and challenges for different stakeholders in each type of educational hackathon.

University perspective. Universities constantly struggle to achieve the economic, technological, and political factors influencing technology development speed. Intensive hackathons can extend core content, such as IEEE/ACM curricula guidelines, without overstressing the curriculum. Hackathons are also a new way to evaluate skills and competencies in a real environment, as Harz did in Germany. This gives students truly meaningful way to test of their skill and ties their experiences to real life.

Hackathons share many goals and objectives with the capstone projects and, thus can provide a good alternative approach to teach students real project skills. As hackathons come in many shapes and sizes, curricula can be planned to include several hackathons with slightly different learning objectives. Repeating basic learning objectives strengthens these skills even more. University-arranged hackathons should emphasize this as a primary outcome.

Another advantage of hackathons is intensified stakeholder collaboration (e.g. with surrounding society and industry). Hackathons, such as those shown above, are well suited to industry- or society-focused projects, in which universities provide valuable contributions to the surrounding environment. In summary, universities as stakeholders get the following positive outcomes from hackathons and code camps a like: ***New course content; New evaluation methods; Intensive implementation of a "project course;" and Close collaboration with other stakeholders.***

In addition to these positive outcomes, universities also face a few challenges as stakeholders that they may wish to learn about and address in earlier stages:

- *Additional effort required*: Compared to theoretical core courses, hackathons require more preparation in terms of logistics, recording outcomes, management, and industry communication;

Students, mentors	Skill to work with a real-world challenge, innovation capabilities, implementation, competition entry
Companies, students	Ability to work on a real-world challenge and operation environment

- *Effect on event participants' health*: Time constraints pressure participants to stay wake to achieve the event goal within the allotted time frame, resulting in less sleep and potential negative effects on participants' health;

- *Effect on study–life balance:* Students have found that it is sometimes difficult to balance study time and personal life, which can lead to less longer-term learning. For example, one student reported: "*As the Code Camps are typically these intensive learning events, time management can get tricky. E.g. I have other duties in life too and I can handle those more easily with more traditional teaching as bunch of separate courses give you more flexibility than one intensive one.*"

- *Free-rider problem:* Students reported that some participants only participated in the course to obtain either credit or a grade. For example, one student noted: *"There is a person in the group who does just the bare minimum."* However, many students also reported that this did not happen when they were able to choose their teammates freely, as people who were known to be free riders were not popular choices.

To address the challenges related to potential negative health effects, teachers should build the program in such a way that students must start work in the morning (and not at 2 pm). Student reflections revealed that if teachers do not regulate the schedule "correctly," students might not participate as intensively as one would expect. Similarly, the free-rider problem could be prevented by following the steps presented by Levin [23].

Our experience, however, shows that this effort in organizing code camp decreases with experience and repetition.

Student perspective. All courses should lead to new accumulated knowledge, skills, and competencies, either hard or soft. On the one hand, hackathons can be seen as narrow technology-focused courses that add some soft skills to knowledge of a selected technology. How much can one learn in 24 hours or even a week? This depends on how the hackathon has been planned and implemented. For example, the studied freshman code camp does not have deep technological objectives. Instead, it is primarily designed to help students get a fast track in their studies, while strengthening their self-belief and ability to overcome challenges.

Curricula emphasize real work–life experiments, which are exactly what hackathons provide. Students' customers seek solutions to challenges, and students develop these solutions collaboratively. By addressing industry challenges, the students work with real problems. If a hackathon is used as an evaluation method, then the exam is brought from the exam space to a realistic environment in which students can continue learning.

Hackathons also provide students appreciation and recognition. They can use their projects in their study portfolios and present the results to possible employers. Some hackathons include industry stakeholders who are looking to hire the best students. Students who participate in competitions or industry hacks may even earn

public appreciation for doing well. Thus, students as stakeholders get the following outcomes from hackathons Hard and soft skills; Real-life projects and evaluations; Employment; and New connections in their work–life networks.

Industry perspective. In the first years of hackathons (1st generation), industry representatives gladly collaborated with universities in educational hackathons, as this approach ensured that graduating students had the skills companies needed. Thus, hackathons were beneficial for both companies and universities, and they linked students to real-world challenges. For industries, these events were an inexpensive way to connect with prospective employees. For example, one industry stakeholder noted that the "*[h]ackathon opens our eyes to new ideas and thinking directions," reflecting that "[w]e will 100% definitely participate in the event next year. The event was a kick start for new product development efforts*" (CEO from a big company). Similarly, the CEO of another software company stated, "This was a fun experience. Academia arranged the operative part, we offered our field expertise, and at the same time found two new potential recruits!"

The field changed with the rise of industry hacks (2nd generation), as companies started to organize hackathons themselves or through commercial organizers. At the same time, the educational value of such hacks has decreased, and companies have begun to collect value through new business solutions instead of student ability. Industry hacks no longer work as a way for students to find employment; thus, universities hardly ever have a role in these (except through, for example, case study researchers).

The following are the outcomes industry as a stakeholder gains from code camps include: New solutions, services, business-to-business networks, publicity, and brand marketing; Close collaborations with universities; and Links to new employees.

In addition to these positive outcomes, we discovered that industry as a stakeholder should also attempt to address a few challenges, as follows:

Uncertainty regarding the solution and potential cost: Industry participants found that code camps usually involved uncertain outcomes and costs (i.e. time spent). For example, one industry participant reported, *"You never know: Do you get any valuable solutions in the end (especially in extremely challenging hackathons with high-level business goals?"* Similarly, another reported that *"costs are always there (e.g. time spent in the event), but results cannot be guaranteed."*

Society and citizen perspective. Public organizations and society in general have partly taken on the original role of industry in providing real-world challenges for the universities and students. However, society cannot provide technical knowledge to universities in the same way that industry did.

The outcomes that society as a stakeholder gets from code camps include: New innovations; Visibility for new technologies; and Citizen Participation. Additionally, both hackathons and code camps illustrate most essential features of collaborative learning, which has been shown to be highly beneficial in software engineering education and learning in general. These events include all the essential aspects of cooperative learning, as defined by Johnson and Johnson [16]:

- Perceived positive interdependence. Team members are responsible to one another for the shared success of the team. Members can support one another to achieve the goals (e.g. by sharing skills).
- Considerable promotive (face-to-face) interaction. With continuous collaboration and support to achieve learning goals, student teams can achieve better overall outcomes.
- Perceived individual accountability and personal responsibility to achieve group goals. Through experience, teams quickly judge the "fairness" of the workload and assign responsibilities to everyone.
- Frequent use of relevant interpersonal and small-group skills. Task divisions must be negotiated to address the overall team goal within the theme limitations of the event.
- Frequent and regular group processing of current functioning to improve future group effectiveness. Typically, the ends of events involve team reflection on processes. Optimally, the facilitators also meet the team, give feedback and guide them through reflections.

6 CONCLUSION

To respond properly to the growing global business environment, education requires continuous changes to improve the students' workforce readiness. This paper studies the concept of intensive collaborative learning events i.e. hackathons and code camps, in order to show how the events can be harnessed within software engineering education to teach the necessary skills and competences for the students. The study also identified wide range of stakeholders that can be part of ensuring the success of this concept of intensive social learning environment. Stakeholders within this concept of social learning environment can extend to many levels of society including students, teachers, educational units, industry and cities. Within these extensions, different stakeholders can expect different outcomes. We have identified several of them and listed in the chapter for as per stakeholder. As conclusions, it can be stated that most typical outcomes are 1) new networks and connections, 2) skills that surface (either achieved new skill or a skill that was tacit) and 3) someone receiving either new solution or ideas.

The results indicate both concepts fulfill the needs of i) capstone project (i.e. adjust teaching to quickly changing world and teach collaborative work); ii) students (i.e. learning hard and soft skills to perform better software engineering activities); iii) society (stimulate the discussion and validate the concept to solve the real problem within the society). However, there is evidence that results are always depending on the level of i) input the students put in; ii) effort the industry puts in participating truly in this collaboration; and iii) effort the academia puts in connecting students and industry together without forgetting the role of academia as provider of neutral ground for the industry to "meet the new ideas" and students to take new challenges. Also, the intrinsic motivation of all participants to learn something new is an important key factor for

providing a ground for a successful event. This means, whoever is building events like this, should have clear objectives from the beginning (i.e. what are the things the participants get from the event). Without this gain (intrinsically or extrinsically motivated), one does not put as much effort into the collaboration as one could.

Our experience on the first aim "In what level can hackathons be used in software engineering education to teach the necessary skills and competencies for the students." shows that we need to consider the level of participants. For freshmen level, we teach the skills and competencies for following studies. In the middle of the studies, we teach the skills and competencies for understanding new technologies and in final years teaching is pushing especially the skills and competencies needed in work life (e.g. activating your customers, promoting your solutions, considering user experience and so on). To summarize, work here is based on our own experiences and lessons learned: i) can be used at different levels of education to help the professional skills being developed at that time; ii) however, does not give a general taxonomy for these kinds of events. As future work, we recommend continuing this classification work and working with the diverse stakeholders to establish recommendations and best practices for their participation. The main limitation of the work is that hackathons, code camps, and other intensive collaborative events are so diverse in arrangements, participants and goals that some of the events escape this overview and classification.

ACKNOWLEDGMENTS

We would like to thank all the participants of code camps. The work of the fifth author was supported by the Ulla Tuominen Foundation.

REFERENCES

[1] Banjac, M. 2017. E-participation as a technology of citizenship. *Teorija in Praksa*. 54, 1 (2017), 73–91.

[2] Beetham, H. and Sharpe, R. 2007. *Rethinking pedagogy for a digital age: Designing and delivering e-learning*.

[3] Belbin Team Roles | Belbin: 2016. *http://www.belbin.com/about/belbin-team-roles/*.

[4] Briscoe, G. and Hon Xuan Jia 2014. *Choreograthons: Hackathons for Dance Composition*.

[5] Briscoe, G. and Mulligan, C. 2014. Digital Innovation: The Hackathon Phenomenon. *Creativeworks London*. 6 (2014), 1–13.

[6] Bruns, A. 2007. Produsage. *Proceedings of the 6th ACM SIGCHI - C&C '07* (New York, New York, USA, 2007), 99.

[7] Chesbrough, H.W. 2003. *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Harvard Business Review.

[8] Chounta, I.-A., Manske, S. and Hoppe, H.U. 2017. "From Making to Learning": introducing Dev Camps as an educational paradigm for Re-inventing Problem-based Learning. *IJETHE*. 14, 1 (Dec. 2017), 21. DOI:https://doi.org/10.1186/s41239-017-0061-2.

[9] Davies, W.M. 2006. Intensive teaching formats: A review. *Issues in Educational Research*.

[10] Decker, A., Eiselt, K. and Voll, K. 2015. Understanding and improving the culture of hackathons: Think global hack local. *Proceedings - Frontiers in Education Conference, FIE* (2015).

[11] Dunlap, J.C. 2005. Problem-based learning and self-efficacy: How a capstone course prepares students for a profession. *ETRDd*. 53, 1 (Mar. 2005), 65–83. DOI:https://doi.org/10.1007/BF02504858.

[12] Gillies, R.M. 2006. Teachers' and students' verbal behaviours during cooperative and small-group learning. *British Journal of Educational Psychology*. 76, 2 (Jun. 2006), 271–287. DOI:https://doi.org/10.1348/000709905X52337.

[13] Gokhale, A. a 1995. Collaborative Learning Enhances Critical Thinking. *Journal of Technology Education*. 7, (1995), 22–30. DOI:https://doi.org/10.1007/978-1-4419-1428-6_910.

[14] Ikonen, J., Hamalainen, H., Alaoutinen, S., Heikkinen, K. and Porras, J. 2009. From tacit to acknowledged knowledge. *2009 EAEEIE Annual Conference* (Jun. 2009), 1–6.

[15] Johnson, D.W. and Johnson, R.T. 2002. Learning Together and Alone: Overview and Meta-analysis. *APJE*. 22, 1 (Jan. 2002), 95–105. DOI:https://doi.org/10.1080/0218879020220110.

[16] Johnson, D.W. and Johnson, R.T. 1999. Making cooperative learning work. *Theory Into Practice*. 38, 2 (1999), 67–73. DOI:https://doi.org/10.1080/00405849909543834.

[17] Johnson, P. and Robinson, P. 2014. Civic Hackathons: Innovation, Procurement, or Civic Engagement? *Review of Policy Research*. 31, 4 (Jul. 2014), 349–357. DOI:https://doi.org/10.1111/ropr.12074.

[18] Julie, M. and David, T. 2003. Engineering Education, Is Problem-Based or Project-Based Learning the Answer. *Rapid Prototyping Journal*. 8, 2 (May 2003), 116–125.

[19] Kilamo, T., Nieminen, A., Lautamäki, J., Aho, T., Koskinen, J., Palviainen, J. and Mikkonen, T. 2014. Knowledge transfer in collaborative teams: experiences from a two-week code camp. *Companion Proceedings of the 36th - ICSE Companion 2014*. (2014), 264–271. DOI:https://doi.org/10.1145/2591062.2591156.

[20] Knutas, A., Ikonen, J., Nikula, U. and Porras, J. 2014. Increasing collaborative communications in a programming course with gamification. *Proceedings of the 15th International - CompSysTech '14* (New York, New York, USA, 2014), 370–377.

[21] Komssi, M., Pichlis, D., Raatikainen, M., Kindstrom, K. and Jarvinen, J. 2015. What are Hackathons for? *IEEE Software*. 32, 5 (Sep. 2015), 60–67. DOI:https://doi.org/10.1109/MS.2014.78.

[22] Laal, M. and Ghodsi, S.M. 2012. Benefits of collaborative learning. *Procedia - Social and Behavioral Sciences* (2012), 486–490.

[23] Levin, P. 2003. Running group projects: dealing with the free-rider problem. *Planet*. 1835, 5 (2003), 7–8.

[24] Nandi, A. and Mandernach, M. 2016. Hackathons as an Informal Learning Platform. *Proceedings of the 47th ACM - SIGCSE '16* (New York, New York, USA, 2016), 346–351.

[25] Okamoto, T. 2004. Collaborative technology and new e-pedagogy. *IEEE ICALT, 2004. Proceedings.* (2004), 1046–1047.

[26] Palacin-Silva, M., Khakurel, J., Happonen, A., Hynninen, T. and Porras, J. 2017. Infusing Design Thinking into a Software Engineering Capstone Course. *2017 IEEE 30th (CSEE&T)* (Nov. 2017), 212–221.

[27] Porras, J., Heikkinen, K. and Ikonen, J. 2007. Code Camp: A setting for collaborative learning of Programming. *Advanced Technology for Learning*. 4, 1 (2007), 43–52. DOI:https://doi.org/10.2316/Journal.208.2007.1.208-0906.

[28] Porras, J., Ikonen, J., Heikkinen, K., Koskinen, K. and Ikonen, L. 2005. Better programming skills through Code Camp approach. *16th EAEEIE Annual Conference on Innovation in Education for Electrical and Information Engineering, Lappeenranta*. (2005), 6–8.

[29] Puntambekar, S. 2006. Analyzing collaborative interactions: divergence, shared understanding and construction of knowledge. *Computers & Education*. 47, 3 (Nov. 2006), 332–351. DOI:https://doi.org/10.1016/j.compedu.2004.10.012.

[30] Raatikainen, M., Komssi, M., Dal Bianco, V., Kindstom, K. and Jarvinen, J. 2013. Industrial experiences of organizing a hackathon to assess a device-centric cloud ecosystem. *Proceedings - International Computer Software and Applications Conference* (2013), 790–799.

[31] Sakhumuzi, M.D. and Emmanuel, O.K. 2017. Student perception of the contribution of Hackathon and collaborative learning approach on computer programming pass rate. *2017, ICTAS 2017 - Proceedings*. (2017). DOI:https://doi.org/10.1109/ICTAS.2017.7920524.

[32] Tarricone, P. and Luca, J. 2002. Successful teamwork: A case study. *Proceedings of the 25th HERDSA Annual Conference, Perth, Western Australia, 7-10 July 2002*. (2002), 640–646.

[33] The Joint Task Force on Computing Curricula, A. for C.M. (ACM) and I.C.S. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM, Inc.

[34] The Joint Task Force on Computing Curricula, A. for C.M. (ACM) and I.C.S. 2014. Software Engineering 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. February (2014), 1–134.

[35] Ward, D., Hahn, J. and Mestre, L. 2014. Adventure Code Camp: Library Mobile Design in the Backcountry. *Information Technology and Libraries*. 33, 3 (Sep. 2014), 45. DOI:https://doi.org/10.6017/ital.v33i3.5480.