How Hackathons Will Change IS Students

Taylor Whitman Poulos
Carnegie Mellon University
tpoulos@andrew.cmu.edu

ABSTRACT
Hackathons, a new type of programming contest focused on full application development, are becoming popular. Many Information Systems (IS) students will participate in them, and their education will be changed in many ways because of this new focus. First, hackathons give students the time and motivation to work on projects that are outside the domain of a traditional curriculum, which allows them to focus on emerging technologies and specific domain areas of interest. Second, they foster more casual interactions with companies and encourage entrepreneurship. Thirdly, they change the way students think. Designers are taught to value minimalism and simplicity in their designs. Project management becomes less centralized, and teamwork is highly valued. Finally, hackers are taught to accept risk, valuing speed over clarity.

Keywords (Required)
Hackathon, Information Systems, extracurricular education, entrepreneurship

INTRODUCTION
Hackathons are becoming a staple in the techie college experience. Part programming contest, part maker's fair, hackathons are a chance for teams of students to sit down and make something with the skills they've been learning in their classes. The entirely student-run events last between 24 and 48 hours, and the participants are encouraged to be programming almost the whole time. It isn't uncommon to see hackers sleeping on chairs and couches in the school buildings where they're held, or sitting with empty energy drinks scattered around their workspace. At the end, prizes are awarded to the best projects, with thousands of dollars in cash as the top prize. Partly because of the prizes and partly the caffeine, hackathons jitter with the excitement of a LAN party.

In less than three years, hackathons have been transformed from a handful of 300-person programming competitions put on by schools like Penn, Columbia, and Princeton to 1,500 hacker competitions. The organizations that run them, which are almost entirely student-based, raise six figure budgets from startups and big tech companies, like Facebook, Twilio, Dropbox, and Google. Carnegie Mellon University was one of the first schools to run a hackathon. Called TartanHacks, it's run every spring by a team of students called ScottyLabs.

Because hackathons have become popular only recently, it is difficult for current IS students to see the ways hackathons change our college experience. I, personally, have participated in over 10 hackathons, and assisted in the organization of two. This paper is based on my personal experience as well as on my discussions with my fellow hackers. There is currently a paucity of articles examining this subject, and it is my hope that this will inspire other hackers to look at how hackathons are affecting the way we learn, get jobs, and most important, the way we solve problems.

LEARNING
The modern Web is very fast-paced. As a result, it is difficult, if not impossible, for IS professors to keep pace with it. Rather than keeping track of the latest and greatest JavaScript libraries or experimental HTML5 features, most professors teach fundamental concepts, the things that won't change year to year. Although this approach creates developers that will remain effective over many years, it leaves something to be desired.

Many students want to use experimental technologies. Because hackathons are ungraded, students are free to try out high-risk projects without worrying about ruining their grade-point average, and that freedom stimulates an intellectual sandbox of technological creativity. In addition, students are exclusively working on projects that interest them. The only restraint on a hackathon project is that any code you use must be either open source or written during the hackathon (so that everyone starts with the same resources). From this starting point, people move in all sorts of directions. Although a common hackathon project is something of a contradiction in terms (because great hackathon projects should be novel), some examples of projects that I see often include: textbook marketplaces, speech-to-speech translators, Markov Chain Facebook status
generators, and multidevice coordinated displays. Generally, the goal is not to make a clean, robust app. The main goal is to get the thing working, and as such the projects are more of a demo than a releasable project.

Hackathon projects only occasionally form the basis for a longer term project. Instead, they are one-off, quick explorations of a novel idea. This means a few things. First of all, students will graduate having explored more advanced topics, most of which might not be covered in their classes. Second, students will have deeper knowledge of the basic concepts that their classes teach because those concepts will be enforced by more project experience than the traditional curriculum. Third, people will be more adept at starting projects. They will be able to generate and improve project ideas with skill, and they will be less intimidated by starting a project entirely fresh. These are all skills that improve their education, and I would argue that hackathons are a useful educational experience.

WORK

Hackathons will also change the way students interact with companies. In the past, students had to attend a job fair or information session to interact with a company. Hackathons give companies a direct link to students, something that was simply not done before. Sponsors are routinely given access to student’s résumés, majors, and contact information and are invited to walk around, serving a dual purpose as technical mentors and recruiters. In some ways, this represents a much more natural form of interaction than a job fair. By asking a question like, “What are you working on?” a company can gauge a student’s interest in its domain, identify a student's strongest skills, and figure out where he or she might fit in at the company. Students also have the chance to approach companies and even use their products in projects (which often leads directly to an interview). More fundamentally, when a company sponsors a hackathon, it is directly sponsoring an event that the students love rather than supporting a school’s career center.

Finally, hackathons encourage students to look beyond the traditional job system. Entrepreneurship language is common at hackathons. Speakers often encourage students to take their projects to incubators like Y Combinator, and most offer startup themed prizes. People often see the hackathon process as an abbreviated startup experience and assume that hackathons prepare students for the world of entrepreneurship. Although starting a company is certainly a longer and more complicated experience than attending a hackathon, I expect that many students will pursue a startup because of their hackathon experience.

THOUGHT PROCESS

Finally, it’s important to remember that by participating in hackathons, students are changing the way they think as designers, project managers, and programmers.

Design

Design at a hackathon is an exercise in minimalism. Projects need to be created in 24 hours and demonstrated in less than a minute if they are to win prizes. Extraneous features are quickly cut, and effective teams quickly focus on the most minimal product that solves their problem. Projects that win hackathons are quick, fluid, and brilliantly simple. Great hackathon designers are generally skilled programmers as well because they are generally expected to write the interface. Very few teams make Photoshop mockups of their product beforehand; in fact design is often done on the bus to the hackathon or spontaneously throughout the event.

Project Management

Project management at a hackathon is often a shared responsibility. Most hackathons limit teams to four people, which is considered the optimal number for fast-paced development. The time frame limits software development methodologies somewhat. Although there is not enough time to complete more than one sprint, most developers tend towards an AGILE inspired system; first achieving a working product, then adding features until they reach a minimum viable product. For other projects there is little time for anything other than waterfall methods, as all of the components must be in place before the desired behavior will be achieved. Experimentation with different methods and strategies is common, and many students come away from a hackathon having learned something about how to manage a project. Motivated students can use hackathons as an incubator for project management skills, battle hardening their ability to work under severe constraints.
Programming

Finally, it is important to comment on the way hackathons nurture participants’ programming skills. The main difference between classroom programming and hacking is that at hackathons programmers are free to use “bad” technique as long as the code gets written faster. Code quality is less important than getting things working, which means that security and maintainability are often the first things tossed aside. This is obviously different than development as seen by schools or corporations. Code written at hackathons would be considered unusable in most corporate environments. Really, programming at a hackathon is more similar to prototyping than real “development.” However, hackathons make programming more creative. Programmers are given the freedom to hack things together however they want, and that encourages analytical thinking beyond the classroom level, where the environment is carefully controlled and problems are well-defined. This kind of creative freedom is dangerous. Oftentimes, hackathon projects are buggy or poorly architected because their developers got in over their head. It’s important, though, to remember even failed projects are educational, and exercise skills. Although hackathons may not produce careful, perfect code, they do produce better programmers.

CONCLUSION

In conclusion, I believe that hackathons are changing IS students for the better. Because of hackathons, the IS graduates of the future will not be green and inexperienced. Rather, they will be battle hardened by their hackathons experiences, used to ideating quickly, working all night, and starting lots of projects. However, without guidance, I expect that they will be less careful and more willing to take risks to get a project done quickly.

In the future, I expect to see hackathons become an even bigger part of student life. I believe that hackathon attendance will become an expectation rather than an extracurricular, and that elements of hackathons will be incorporated into official school curriculums. As hackathons get bigger and more legitimate, I believe that it will become more and more important to remember their roots. Hackathons are about making things for the sake of making them, not for professional advancement. The spontaneity of hackathons makes them powerful, and they represent an exciting way to improve our trade.

ACKNOWLEDGMENTS

I thank Jeria Quesenberry for reviewing an early draft of this paper. I also thank the anonymous reviewers for their comments, and ScottyLabs for giving me the opportunity to help out with TartanHacks.

APPENDIX 1: MAJOR HACKATHONS AND THEIR ORGANIZERS

<table>
<thead>
<tr>
<th>Hackathon Name</th>
<th>School</th>
<th>Organizing Body</th>
<th>Hackers (Approx.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TartanHacks</td>
<td>Carnegie Mellon</td>
<td>ScottyLabs</td>
<td>300+</td>
</tr>
<tr>
<td>PennApps</td>
<td>Penn</td>
<td>Dining Philosophers</td>
<td>1,000+</td>
</tr>
<tr>
<td>HackMIT</td>
<td>MIT</td>
<td>HackMIT Committee</td>
<td>1,500+</td>
</tr>
<tr>
<td>HackNY</td>
<td>(unaffiliated)</td>
<td>HackNY</td>
<td>400+</td>
</tr>
<tr>
<td>MHacks</td>
<td>UMichigan</td>
<td>MHacks Committee</td>
<td>1,200+</td>
</tr>
</tbody>
</table>