A New Place to Work and Play:
Play Labor and the Production of the New Worker-Subject at Hackathons

Audrey Le

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee of
the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2017

ABSTRACT

A New Place to Work and Play

Audrey Le

Since 2012, hundreds of companies have poured thousands of dollars into hackathons – finite events where creatives come together in small teams to design, build, and demo a new product of feature. The spectacle of the hackathon engages participants in a number of things: a transgressive ethos, disciplined play, and hacker's literacies (Santo 2011). Based on my dissertation fieldwork at seven hackathons in three industries (journalism, healthcare, and e-government), I explore various types of play labor (Terranova 2000) based on the performances of eight teams. I show how teams creatively manage their peers' affective and intellectual labor, and negotiate what appear to be industry-specific preferences for different technologies. In the process of competing for status and recognition, they engender distinct forms of play labor and making do. Hackathon participants directly embed resistance in their designs; some learn how to learn (Bateson 1972), giving them a strategic advantage over other classes of workers. They embody the characteristics of the new worker-subject required in the digital economy, as mutable, playful, and rapid.

TABLE OF CONTENTS

List of Tables

List of Figures

Glossary

Hack: A solution to a problem. Code in a computer program that has been modified in a creative and clever way. Often likened to tricks, pranks, and security breaches.

Hackathon: An event held within a finite amount of time (typically 24 or 48 hours), where people come together to solve problems and build these solutions overnight.

Hacking: Creative problem-solving, or the production of hacks. Hacking involves figuring out how to manipulate existing code to make systems do something that they were not intended to do. It also refers to the act of flipping the logic of a practice unto itself, and creatively embedding a hack in an existing community of practice.

ACKNOWLEDGMENTS

To Graham Jones, without whom this research would not be possible. He sponsored my fieldwork at MIT and helped me explore themes in my data early in the writing stage.

My dissertation committee was diverse and fundamental to making the connections in this dissertation. My advisor, Herve Varenne, ignited my curiosity and passion for literacy and technology early in my graduate school career. I am grateful for his support during my visits at other universities to conduct my fieldwork and finish my coursework. Brian Larkin introduced me to the scholarship on digital labor during my exams. His time and feedback were essential to my success. Nicholas Limerick's in-depth knowledge of issues in linguistic anthropology bolstered my transcription methods and theoretical understanding of language as communicative action. Brian Boyd's versatility and availability on earlier drafts helped me develop my understanding and critique of actor-network theory in my studies of the social production of technology.

To my best friends: Becky, Yi Ou, Crystal and Caitey for their faith in me and the strength they have given me to finish graduate school.

Many thanks to Jacob Schlosser, Ashwin Murali, Thomas Bourgeat, Dan Hussain, Jihii Jolly, Ying Tan, Michelle Skinner and Nhon Ma for supporting me in all manners of engineering expertise, accommodations, and for being exemplary models of entrepreneurship and community engagement. Thank you for taking such an interest in my work and providing me with an emic perspective on Boston's ecosystem as backdrop for hackathons.

Guangtian Ha, Tristan G. Brown and Max H. Dunitz offered me their infinite support while I wrote this dissertation. Their disciplinary differences pushed the boundaries of my analyses and helped me improve my craft on the way, too. Dan Souleles, Jennifer Van Tiem and Michael Scroggins looked out for me from the beginning of my graduate school career, and were always

available to discuss some of my favorite authors.

To the people who helped me develop my voice as a scholar and educator: Beth Semel, Shreeharsh Kelkar at MIT, for their help in documenting my fieldwork; Anna Filipova, Arnab Nandi and the research group at the CSCW 2017 Workshop on Hacking and Time-Mediated Events for nurturing my interest in hackathons; Reut Harari, Joel Rozen at Princeton for offering direction; with support and friendship from Jordan Wright, Sarah French Brennan, Laura Hudson Meghan Chidsey and Amiel Melnick at Columbia University.

When I was not knee-deep in social studies of science and technology, Nonie Lesaux, Marjorie Siegel, Christina Dobbs were major inspirations as teacher educators. They have opened my eyes to new literacies, and shared many resources at the Harvard Graduate School of Education and Teachers College available during my pilot.

I would like to thank Gabriella Coleman and Tiziana Terranova for pioneering the study of hackers and digital labor and their input while I was initially exploring that space. I am grateful to Paul Kockelman for his prolific scholarship on the place of information theory in anthropology, which prepared me for this study.

Amina Tawasil, Louise Beryl, Ernesto Martinez and Janny Chang – thank yuo for your mentorship!

To my father, sisters and mother for their unconditional love. My husband is the most heartwarming individual and was most supportive of my academic journey.

Introduction

My first hackathon happened in the spring of 2013, while I was visiting at the Harvard Graduate School of Education. I received an e-mail from the HIVE, an education technology student club on-campus, inviting all Harvard student teachers to join students at the Massachusetts Institute of Technology (MIT) for a weekend of ideation and designing innovative solutions to both chronic and emerging issues in today's education market. The participants at the MIT Education Design-a-thon identified themselves as mostly engineers and developers. In less than two minutes, they pitched what they perceived to be a creative technical solution to an educational problem, calling for the kinds of technical and educational expertise they lacked but would need to get feedback on their prototype. Whirling my way through several self-forming teams, I was eventually invited to join a chemical engineer, a developer and a product manager based on our shared interests in a market for a mobile platform for teacher communication. I shot a video for showing how a mobile app filled a need in the teacher market in Rwanda, while our developers feverishly coded the user interface and backend late into the night. At another hackathon, I framed my first use-case for a future Laptop per Child project and mapped the user experience of an archetypal teacher. Watching teams at work, I learned a little about the thought-process of the different fields of engineering, what coders of different abilities valued, and what made technologists tick. When we were not killing our experiments, my teams would improvise based on information about the judges, and debated what there was of value in a barely functional app. We had to agree on which features to keep and which to cut. Some would drop out at-will and join another team, or go home.

A hackathon is an event held within a finite amount of time (typically 24 or 48 hours), where creatives get together in small teams to design, build, and demo a new product or feature. Hackathon participants fall under the category of tinkerer in Coleman's (2014) typology of hackers; they, for one, are not the famous hacker figures behind security breaches. They use public datasets and open-source software to quickly build the beginnings of an app, and must be ready to demo a proof of concept to an audience made up of businesses and nonprofits within a short period of time. Since 2012, the number of hackathons has grown to 150 worldwide every year, the majority of which take place in the US (Major League Hacking 2017).

This dissertation describes what actually happens inside hackathons: moments of consensus and dissociation, and my discovery of the mischief and playful resistance at these events that make up the sociotechnical relations of digital production in industry. I reference at least three typologies in this dissertation – of play, of hackathons, of hackers, on which I draw not for the sole purpose of citing them, but to frame my own typology of play labor. Play labor can be understood as a hierarchy of consequentiality in relation to capitalist modes of production, whereby immaterial labor (Lazzarato 1996) – labor which produces the informational and cultural content of the commodity and increasingly relies on forms of cybernetic control, instead of rather deterministically shaping goods is really forming subjects.

What people do at hackathons is called "hacking" – a form creative programming – that involves writing sets of instructions for a machine to perform. Code practically never does what its author intends it to do. There lies the magic of hacking: one sometimes cannot explain how a particular outcome is produced. Hence, any production is inspired by certain elements of play and uncertainty. The hackathons in this study can relate to a spectacle, which on the one had, ties in its own loose ends – this dissertation is essentially about people working hard behind the scenes to

showcase what technology could be and concealing what did not work that day. On the other hand, the spectacle also functions to communicate the unimaginable (Dean 2012), by drawing on its audience's capacity to engage in leisurely forms of labor – "play labor" (Terranova 2000) – to reflect and simultaneously create new experiences around technology.

My ethnography of hackathons would set forth a precedent in Marxist theorizing of capitalism. It clearly shows that one cannot compare the capitalism of the early industrial revolution period, in which the figure of the engineer in Anthony Wallace's (1978) *Rockdale* threatens to undo what the capitalist had paid for, with the post-Fordism of the late twentieth century, for which the increasing reliance on information technologies to do work has called for a growing class of entrepreneurs to manage their social relations of production (Lazzarato 1996). Each industry that organized hackathons in this dissertation showed different preferences for technology and correspondingly, resorted to different types of play labor in order to solve different kinds of problems. The assemblages of tools, people and institutions that made up the hacks in these six chapters indicate that something different was at stake every time. Like Latour (1986), I was able to write about personal values without requiring the reader of these pages to keep track of who these hackers were, and contribute to the study of play's relation to labor a new understanding of the kind of worker-subject is required of this digital economy.

An Anthropology of Play Labor

Play labor was a term invented by Tiziana Terranova (2000) during the dot-com bubble, for which much speculation about the possibilities associated with the commercialization of the Internet brought about a new tradition of scholarship on digital labor – the acceleration of forms of collective intelligence that were more affect-driven and for which a growing reliance on the

axiomatization of the body (Deleuze 1992) came to replace certain jobs in the increasingly automated industrial sector. Play labor concerned the importance of leisurely, affective labor that fuels the growth of technology-enabled sectors of business, among them journalism, healthcare, and e-government.

This dissertation is not about a specific region, or broader contexts of post-industrial production. I came to write my research in this current form because I disagreed with Terranova's argument. My research shows that play labor could indeed be separated from capitalist flows, but still imagined in relation to it. I am particularly interested in a typology of play labor, and who might thrive under duress at these fast-paced events. Who might excel at hackathon-specific types of creative problem-solving and what kinds of skillsets do they strategically display to manage the cooperation? Because hackathons do not stray too far from a deferred wage, I go as far as arguing that some hackathon-related forms of play labor, characterized by the central ability to learn how to learn as some success stories tell us, gives a class of digital workers i.e. programmers, designers and entrepreneurs, a strategic advantage over forms of degraded labor.

My first introduction to play labor came from hackers' testimonies I collected here and there, on YouTube, blogs and interviews. Fredrick Wagner, the lead developer of the Anti-Corruption University hack at Hack for Democracy, said so in his follow-up interview: "I, I completely, attribute my ability to subsequently find a job here, to my experience at the hackathon, like having such a positive, collaborative, constructive experience there, that's what gave me the confidence I needed to go out and interview and land a job." For Fredrick, participation at a hackathon can build confidence; the analogous relationship between hackathons and job preparation clearly establishes participation as a form of rentable labor, and the event as a platform to *practice* marketing one's assets later. An organizer at MIT Hacking Medicine, Nicholas, added:

"hackathons got me to test the limits of my knowledge of design very quickly." Andrew, a biomedical engineer at the Athena Health MDP hackathon in Boston, explained: "Hackathons create a sandbox for us. The same way video games do."

These statements told me that something else was at stake at these hackathons, which had more to do with:

- Testing one's limit and assumptions;
- Mentorship, meeting people who can take your work to the next level;
- "Sandboxing," and "drills" - a safe place to try new roles and new projects.

In other words, hackathons take participants out of their ordinary work lives – that is one type of play. They give them resources and talent to experiment and learn something about *themselves* – the sandbox is another. We will see in later sections how different types of play labor differentiate participants as digital worker-subjects.

To make this argument, I draw from two kinds of scholarship about play. One tradition, going back to Bateson (1972), along with sociologists of play whose typologies are still frequently cited among game theorists: Huizinga, Caillois and to a certain extent, Sutton-Smith, posit that play is an activity of little consequence. This structuralist framework for play helps to understand why the hackathon model of organizing labor appeals to hundreds of participants all over the United Statess, as a self-contained, self-sufficient space to explore design challenges and market alternatives, with relatively few changes to their statuses – only a handful get a job or internship out of participating – or laboring praxis for that matter. Participation at hackathons does not change their job responsibilities but allow them to think differently about a problem.

Those who study play as nonconsequential drew their primarily Durkheimian insights from the skillful, orderly nature of play, while a Marxist tradition studies specific leisurely forms of

digital labor – or playbor – that are subject to being exploited by corporations. The Italian Autonomists, led by Maurizio Lazzarato, Tiziana Terranova and Antonio Negri, project direct social and economic consequences to forms of belonging on the web, which I discuss presently. Virno, for instance, reworked the possibility of socioeconomic classes in this new dominant system of economic production and consumption. There is no unity of class, only multitudes[1], which need worldly institutions to metabolize their ambivalence and oscillation rather than unilateral decisions. Virno located in the faculty of language the virtuoso's ability to learn and his access to self-reflection, which he argues are potentials for political action. Far from being machina machinarum[2], virtuous conduct is an end in itself.

While Virno's politics fragment workers' subjectivities, Lazzarato argued instead for the rise of a brand-new worker-subject to manage social relations: the entrepreneurs. This entrepreneurial class manages the production of affective and cognitive commodities outside of a traditional wage-system, also called "immaterial labor". Terranova (2000, 104) would argue that the fruit of this collective cultural "free labor" on the Internet has voluntarily been channeled and controversially structured within capitalist business practices. My contribution to this discussion on play labor, however, is a full synthesis of all forms of laboring at hackathons, sometimes directly tied to cognitive capitalism, sometimes only a form of making do. A typology of play labor makes it just a little bit easier to pick apart these notions as they happen in-situ.

Outline of chapters

My goals for this dissertation are to describe the forms of play labor that are specific to the organization of a hackathon. The first and second sections cover affective and leisurely forms of

[1] This is a recurrent theme in political philosophy, which Virno bases on Negri and Hardt's (2004) notion that the multitude does not need to enter a social contract with a sovereign political body to exist. Virno (2004, 40-42) calls it a publicness of action without a public sphere.

[2] Virno (1996) goes against the Hobbesian notion that the state is a techno-rational machine.

intellectual labor that progressively become more disciplined in the third section. I show how this disciplinary worker-subject can compete at a hackathon from an ability – individual or collective – to learn how to learn, the varieties of hacker literacies at hand, and disciplined play. Later, I contextualize the free labor of hackathon within the digital economy.

Chapter 1 describes how hackathons organizes this labor. Labor at hackathons differs from the structure of everyday work in the digital economy. The model specifically aims to remove participants from any corporate control of their labor, or ownership of their ideas. Depending on who is invited, talent and resources were temporarily all available in one space to give feedback and more quickly get projects off the ground. The spectatorship of hackathons allowed participants to spontaneously visit the technical side sitting next to them and watch how a technology was assembled. They learned from watching, talking and interacting inform their play. I later describe the actor-networks of judges, organizers and sponsors as one microcosm for one industry, that is, several affinity-based communities of practice (Lave and Wenger 1991). The evaluations of hacks addressed the labor-power of hacker teams by desiring one that reflects the needs and aspirations of the industry.

Section II, consisting of Chapters 2 through 4, is an important overview of the types of play labor that emerge at hackathons. Each chapter covers two teams – one from each of two hackathons in the same industry. Examples from the transcript of their collaboration are provided, and I analyze how one type of play labor within the group's context responded to the unique constraints of the technology they hacked. My use of comparative case studies helped me identify industry-specific preferences for certain technologies, but it is the conversation analyses that allowed me to delve deep into the struggle for work in these industries and how these tensions are negotiated to reimagine the way people work. More importantly, some of these examples demonstrated how play

labor helped teams manage their cooperation.

Drawing on three cases of more disciplined play, Section III focuses on the processes of learning how to learn – deutero- or "double-looped" learning. Deutero-learning was a concept advanced by Gregory Bateson in his studies of Balinese ritual, cybernetics and animal-human communication. Chapter 5 uses this concept to describe forms of disciplined play. Finally, Chapter 6 is a case study involving one team managing various semi-proficiencies in different programming languages. Developing one's hacker literacies, one learns the underlying theory behind a program, which in the case of one team helped it navigate uncertainty in a multilingual codescape. Writing the code was furthermore a playful process for that individual in light that some of the problems that arose from the materiality of the hack.

Weaving all three sections into one coherent argument, I reprise Virno (is play a form of political action?), Terranova (is it a form of leisurely activity tied to late capitalism?) and Lazzarato (does it require a special class of workers?) alongside Bateson (is it non-serious?): If the hackathon simulates the forms of play labor that characterize the new disciplinary worker-subject, as fast, mutable and cooperative; and if their success, and by projection, their ability to labor in the digital economy depends on their ability to learn how to learn from intelligent systems,[3] this ethnography of work at hackathons complicates their arguments. Play labor can challenge fragmented worker subjectivities through competition and extravaganza and improve their skills through reflection. Hackathons in turn reflect a real struggle against work; group participation runs the full gamut of

[3] Bateson (1972) proposed the concept of deutero-learning, or learning how to learn, which involves living organisms, systems and organizations grasping the underlying theory behind a pattern of behavior [the program]. Zeynep Tufecki, alongside Lucy Suchman (July 6, 2016 interview, https://charlierose.com/videos/28424) said of the new developments in machine learning: "there's a way to audit these systems to catch the kinds of biases that creep from the data and understand what's going on. But with the new machine learning systems, even the programmers may not understand and often do not, in fact that's by design, that's the power of the algorithm." Hence, in his daily activity of writing code, the programmer is creating and learning how their own machines learn from data. Suchman cautions against using our preconceived notions about human learning to study machine learning, observing that the programs perform "pretty rote work."

making do, resistance and virtuous conduct that this new class of workers manifests through their personal values. Working around social and technical constraints during hacking sessions, the play labor of the multitude depending on its flexible formation shows different avenues for political action. Many assemblages in fact fail and break down, thus challenging Latour's actor-network theory.

Methods and Social Context/Setting

Hackathons in the US economy

Research into hackathons unearths different histories and pathways for the communities that organize them. I write about industry hackathons, organized by communities of professionals who are inspired by the playful ethos of hackathons and use its form to reimagine the way they work, with different aims: the Hacking Journalism series were a professional development opportunity for data journalists to develop new tools to produce multimedia news stories. The MIT Hacking Medicine hackathons focuses on the business element of producing apps, and winners were encouraged to launch a new company. Conversely, company hackathons are hosted by the world's largest technology companies with the goal to hire the most adept at solving their challenges at the end of the event. Company hackathons have also been used to optimize the sharing of labor and expertise across departments, and limit redundancy so that "no one steps on each other's foot" while working on related parts of a large-scale project. Finally, hacker cons such as DebCon are intensive building sessions, where participants interested in similar open-source projects can meet, collaborate and celebrate their community. The annual edX hackathons have similar to DebCon, and are developer-centric (Coleman 2010). Otherwise, government hackathons in my study recruit technical help on existing projects.

Starting in the mid to late 2000s, hackathons became significantly more widespread. Companies and venture capitalists either saw them as a way to quickly develop new software technologies, or to locate new areas for innovation and funding. A few figures exist to boost their reputation, to be gleaned after much effort from press releases, university websites and blog entries by participants. Hacking Medicine facilitated over 70 hackathons across a dozen countries and multiple US states in the last five years. As of 2015, these hackathons were rumored to have

directly contributed to the formation of fifteen companies that had raised over $90 million in financing (MIT Innovation Initiative 2015).

In an introduction to DevFest[4], an associate professor of computer science at Columbia University, Chris Wiggins, told the story of a time he teamed up with Evans Korth at New York University to host the first university hackathon in 2010. In his interview, Wiggins explained that their intent in hosting the weeklong training and competition had been to prepare their students for the world of the 2008 recession. Hackathons were a response to both internal and external pressures, including fewer job openings, and the development of new technologies that throve off the free labor of communicative networks. New York City saw the most spectacular hackathon at Yahoo! in 2011 (pers. comm., January 13, 2015), at a time the company's stock plummeted, it was bought off by competitors and its leadership experienced high turnover. Companies like Google and Facebook, which regularly host them internally to recruit and train staff were now more willing than ever to shed hundreds of thousands of dollars into hosting these events publicly. To this day, Paypal's Battle Hacks continue this tradition of extravaganza. Among the perks of attending, my friend Joseph Song was amazed to get massages around the clock and all-night access to an ice-skating rink (interview, February 3, 2015). Before they were commercialized, and companies hosting them without knowing what they were doing, its nobler form – hacker cons – had invited hundreds of developers to get together to code their first implementation of UNIX-like operating systems. In 2006, a devoted group of OpenBSD developers crossed the border to Calgary to avoid legal action over the import of cryptographic software into the United States. In just under a week, they had integrated their first protocol with their favorite open-source operating system (Open BSD 2017).

[4] Columbia University's famous week-long hackathon held in the fall.

Reporter Jennifer Elias (2014) published a list of guidelines to help companies decide whether they should host one. Elias recommended aligning the hackathon to the culture and strategic goals of the company hosting it, whether it was to build a long-term community or to tap employees' creativity. Citing Mixbook as the perfect example, Elias described how the company creating a "holistic learning environment" that *mimicked* its culture and served a primary motivation to network with potential new recruits. A friend at Rdio noted the same use of her company's hackathon (Ariel, pers. comm., January 1, 2015).

The main purpose of hackathons, if any could be cited, is to provide people with opportunities to collaborate and see whether these people would want to work together on a separate commercial project later on (Stoltfuz 2015, 1). F-Secure (Raatikainen et al. 2013) has written extensively about the hackathon from an internal company perspective, and lists the main challenges for engagement within teams from the five hackathons they hosted, including: lack of ownership of the project, language barriers in the introduction of a new concept, team dynamics and technological barriers in the validation of a platform for third-party users. Likewise, these barriers affected the performance of teams in my sample. Most hackathons report no sense of leisure beyond the opportunity to work among like-minded peers.

Some hackathons were organized with the mission to broaden participation. Richard et al. (2015)'s survey of Stitchfest, a hardware hackathon that was open to non-coders, shows that event participants valued different experiences: the camaraderie, the sense of pride and ownership, the choice of hardware over software hackathon, and the opportunity to contribute a unique skill.

Even before companies started to host them, the humanitarian sector had used hackathons to fill a gap in technical skills. Drawing from the Random Acts of Kindness model, the World Bank Sanitation and Water hackathons were designed specifically to arrange entry points for developers

with time on their hands to develop tools to address water shortages and sanitation challenges (World Bank 2014). Both the World Bank thematic series of hackathons and the 2014 Make the Breast Pump Not Suck hackathon at MIT sought to involve actual users in the production of something that the latter could use. This model differed from that of other hackathons, which only tried to prove the worth of a novel approach or to identify business opportunities.

The university-based ecosystem for hackathons

The ecosystem for startups includes universities; accelerator programs, which offer a three-month intensive business and design curriculum; access to mentors and investors' network, such as news agencies; city government; and corporations in sectors related to the hackathon.

A hackathon was part of the regular programming of a university or accelerator program. Investors and policymakers often had to wait until the hackathon teams had graduated from an accelerator program to see a direct connection between participation at such events and local economic impact (Bay Area Council Economic Institute 2016). But local economic impact only sometimes captured the benefit of hosting a hackathon. Because the bottom line of a company discouraged experimentation, it became the job of universities to host hackathons and encourage experimentation. Talented, idealist students could compensate for the gap in skills and knowledge available to test new ideas and techniques and do so for free. While universities had a history of partnering with local government to seed patented projects since 1987, the consolidation of venture capital at the university-level did not happen until 2009. The exception would be the University of California System, which had been in the game of taking from pension funds, endowments and other assets, to invest in California startups for the better of two decades (Bay Area Council Economic Institute 2016). In 2017, the year of this dissertation, university incubators and accelerators have multiplied, boasting generous metrics (Table 1).

Table 1 – Universities Investing in Student-Led Startups

University	Venture Fund	Investment	Portfolio Companies	Employment Growth
University of California	UC Ventures (20 incubators)	$250 M in Sept. 2015	622 active out of 1,267 since 1987	38,832 jobs for $16.2 B recent aggregate revenue
University of Michigan	Wolverine	$2 M since 1997	22 companies since 1997	N/A
Massachusetts Institute of Technology	Engine	$150 M in Oct. 2016	TBD	TBD
New York University	Tandon Incubators	$145 M raised in capital	34 companies since 2009	1,260 jobs for $352M local economic impact; ($92M est. tax revenue)

Sources: Bay Area Council Economic Institute 2016; The Washington Post 2015; MIT News 2016; New York Economic Development Corps 2014.

Noteworthy is the type of university launching an incubator and hosting hackathons, namely Ivy Leagues and public universities with high-ranking engineering programs. MIT and NYU have both attracted investments from local venture capitalists[5] and angel investors[6] to go towards Tandon and Engine, their new accelerators. The University of Michigan, conversely, had relied on a single private donation to fund its own programs since 1997. Smaller institutions of higher education like Virginia Tech once managed its $15 million in venture funds through a foundation in 2013, but its alumni are now forming an investor's network modeled after the University of Maryland to pool capital in student entrepreneurship (Overly 2015).

Other economic measures are now available. The New York City Economic Development Corps projected 9,000 jobs by 2025 from the partnership between New York University Tandon School of Engineering and clean-tech incubator ACRE. Its Urban Future Lab has already graduated 34 companies since 2009 and outputted a total 1,260 jobs over a six-year period (New York Economic Development Corps 2014). The University of California system, on the other hand, had

[5] Equity financing managed funds pooled from several investors.

[6] Angels make individual donations through a LLC or family investment fund.

a longer run, and since 1987, has invested in 1,267 companies, which produced 38,832 jobs in California and $16.3 billion in recent aggregate revenue (Bay Area Council Economic Institute 2016). With its reputation for inventions, MIT output will probably be on par with these institutions. As of 2014, the report estimates, MIT alumni have launched 30,200 active companies, employing roughly 4.6 million people, and generating roughly $1.9 trillion in annual revenues (Roberts, Murray and Kim 2015).

Based on the number of investments made by the largest American universities, we can estimate that the demand for companies to help sponsor these hackathons is also high. MIT was uniquely positioned to host hackathons, but until 2012 – the year the Cambridge Innovation Center was built next door, engineering students would attend math puzzle sessions instead (Ross Goodwin, interview, June 11, 2015). Professors cannot put "hackathon organizer" on their curriculum vita. At best, hackathons qualified as "outreach," but these activities did not often weigh in the decision to grant tenure to any professor. And yet, Chris Wiggins was in fact an associate professor at Columbia University at the time he hosted DevFest; Zen Chu continues to be a lecturer at the MIT Sloan School of Business, and Arnab Nandi has been an assistant professor at Ohio State in the four years he hosted Hack Ohio. At the CCSW 2017 workshop, Nandi talked at length about a "cultural attack" at hackathons, which garnered much acquiescence from fellow hackathon organizers and researchers who were interested in the role of play and learning at the hackathon (Krithika Jagannath, response to conference Arnab Nandi's paper, February 25, 2017). Yet in a separate interview, Nandi explained that he had approached his university about a long-term investment in students and was publishing research on possible causal links between hackathons and student learning to convince the administration to give him more funds. He joked that he had been trying to quit his role as a hackathon faculty organizer ever since the first year, but

that the demand for his expertise in hosting them had been too great (Arnab Nandi, interview, February 25, 2017).

Despite his having to parlay the university's administration, the story behind each hackathon in this dissertation, despite their each taking place at MIT, did not share any connection with the engineering school beyond the affiliation of (1) one journalist organizer who had moved from a longstanding career at *The Boston Globe* to accept a new job as a Research Scientist at the MIT Media Lab, (2) the student club at the MIT Sloan Business School whose experience in organizing healthcare hackathons was valuable to companies in that industry, and (3) the proximity of one other institution of higher education, like Harvard, renting the space from them. One needed at least one core organizer to be affiliated with MIT to be able to reserve the auditorium or any of the event rooms at the MIT Media Lab. One was strategic bout hosting at one's institution, for the $7,000 rental fee could be waived. Because of its connection to the top technical and creative talent in Boston, and Boston's startup ecosystem had been grossing one of the highest local economic impacts[7] in the US (US Chamber of Commerce Foundation 2016), MIT was the first choice for hosting hackathons. Most of the participants, however, did not come from Harvard and MIT. But both universities ranked as two of the largest employers in the state of Massachusetts, and digital labor can be said to concentrate in these two locales.

[7] EDC calculates it from (1) business output (or sales volume), (2) value added (or gross regional product), (3) wealth (including property values), (4) personal income (including wages), or (5) jobs.

METHODS

I studied the contexts for play labor through participant interactions, exclusively analyzing conflicts, what hackers said of their participation in follow-up interviews, documenting the processes by which teams deliberated on the possible social value of a hack, and technical constraints as they were reported in-situ. I drew remarkable speech situations from the transcript of conversations that refined my understanding of the social organization of play labor. The transcripts were produced from audio recordings, in natural settings.

I conducted observations at seven 24-48 hour hackathons, four of which took place at Massachusetts Institute of Technology, two at a company site in the Greater Boston area, and one at a company in New York City. During each event, I audio recorded two teams working from start to finish, for a total of twelve teams at seven hackathons. With support from two fellow MIT anthropology students, I was able to access the video footage for two teams. While I visited with the teams, I took notes on their division of labor, topic of conversation (if applicable), and their activities every couple of hours. The skillset or expertise of each participant was readily available online, via professional networking websites such as LinkedIn, personal websites, otherwise autobiographical blurbs submitted with their application to gain entry into the hackathon (if applicable).

Group Observation Protocol

Hacking sessions generated critical insight into event participants' learning process for building a team. I had everyone in these recording sessions state in their consent forms whether they were okay to be audiotaped.

Based on the performances of twelve teams, I was able to obtain a loose leadership structure that varied by expertise, task and roles. The division of labor indicated the level of

activity each team experienced, and was indicative of the merit the individual and the team saw in the task they performed, which correlated with their ability to execute it. Their individual performances indexed each member's commitments and help situate their solidarities as a group, signaling the value of the different components of the hack that they had decided to build individually. The sample size (twelve teams) was large enough to capture the flexibility of each team in choosing their idea, sticking to it, dropping out and joining a new team over three-hour increments.

Recording Pitches and Demos

Hackathons generally start with one individual pitching an idea for a digital product that could remediate a problem found in a specific industry, and end with the team demo of the prototype. These one- to five-minute presentations figure as either a clever or humorous play of words that allow the audience of community members and individual participants to quickly learn more about trends in production for a specific problem or industry. Recovering these trends allow me to situate the value of any team's work in the context of their interactions with industry at the event.

Pitches are volunteered by all participating teams and do not require active recruiting. Since hackathons are a public event, they do not require permission to audio record either.

Follow-Up Interviews

Hackathon participants at MIT generally benefit from a greater availability of resources to execute hacks; therefore, it was important that I describe the network of actors that support each participant. Interviewing individuals on their prior experience with hackathons, and the resources they used or called on during the challenge generated key insights into the heterogeneity of these networks and how they impact performance at the event. Their life histories gave me the broader

context of the digital economy in which the context of hackathons emerged.

I followed up with at least two members of six teams I had the privilege of observing for a sustained period of time. Figure 1 combines participant interviews with those I conducted with organizers, totaling 42 follow-up interviews.

Hacking Journalism Boston: 1 from Twitter Bingo, 1 from Collater, 6 from Newstritious (8 ppl.)
Hack 4 Democracy Boston: 3 ACU, 2 Townhall, 1 organizer (6 ppl.)
Athena Health MDP Boston: 3 from The Cuff Guys, 1 from Reciprocall, 1 organizer (5ppl.)
2014 Open edX: 2 from Docker, 1 organizer (3 ppl.)
Hacking Journalism NYC: 3 from Video Pizza, 2 organizers (5 ppl.)
Hacking iCorruption: 2 from CampaignCon, 3 organizers, 3 mentors (8 ppl.)
2015 MIT Grand Hack: 3 from Sirona Care, 4 organizers (7 ppl.)

Figure 1 – Interview log

The time I waited until I contacted them varied; sometimes I took one month to write the follow-up questions based on a preliminary review of the materials I collected on their collaboration, sometimes I would meet the participants at other hackathons several months later and catch up then, or run into them in Cambridge or NYC. Waiting as long as 6 months gave enough time for participants to pick up their hack again. Due to the intensity of the coding challenge, waiting it out gave participants time to reflect on their learning experience; I suspected that by then, each individual would have fresher insights into their process as a team and the value each could derive from their participation.

Interviewing each team member typically took an hour of their day, and meeting them one-on-one at a cafe gave each individual free reign to express their opinion of the event and their experience as a team. To my knowledge, only one participant in four teams and separate hackathons continued their hack. 11 out of 33 participants repeated their experience at another

event in the industry, sometimes across industries. Two used their experience to help organize another hackathon, and some of the same organizers for the journalism and healthcare hackathon put up the next one I attended in the industry. The first interview was more structured for the first hour, and as I asked interviewees to clarify what they did for each hack.

Organizer's Meetings

Attending one conference for student hackathon organizers, five club meetings for the largest hackathon and two set-up meetings for two others helped me corroborate my preliminary findings on the organization of the hackathon and its impact on participation. During these meetings, I took notes on their recruiting and sponsorship strategies, which required me to find and interview the organizer in charge of implementing the logistics of it. The content of these meetings was somewhat useful as a way to anticipate the flow of the event, but a comparative analysis of the teams for different hackathons better reflected the patterns of play than the organizers could perceive.

Archival Research

Hackathons are widely publicized events and hackers often use Github to publish and collaborate on code. Blogs reported on the pitches for all teams at the event, the event website listed the winners, and the Hackpad, a platform for project submissions, included the names of the members of all teams and links to the team's Github repository, which facilitated surveys and analytics. Reading forums and Facebook group postings often informed me of software developments related to the hacks I studied, which I used in conjunction with follow-up interviews provided me with a three-dimensional view of the imagined value of the hack.

Research Clearance

I had to get permission from the MIT COUHES to collect data at MIT, a fairly

straightforward – not short – application. Since I was observing public events, I could have easily not filed one except that Teachers College's Institutional Review Board required I do so. Research clearance from MIT permitted filming and recording at any off-site affiliated with the university, including Athena Health, edX and Condé Nast.

After the second hackathon, I realized that following up with participants would be difficult given that they lived in many locales. I therefore spent a month in New York City and a few weeks in San Francisco to attend follow-up hackathons and at the same time, follow up with members of select teams. Several participants I observed at journalism hackathons had a job or found employment in NYC after the hackathon. Boston strives to keep up with software developments in the Bay Area, and many hackathon participants eventually moved there to get venture funding for their ideas.

Demographics

To ensure that participants engaged their peers in industry, I limited the seven hackathons to 48/56-hour coding challenges because they tend to be more inclusive and interdisciplinary than other kinds. Coding challenges are typically announced one month before the event occurrence. Student hackers self-select by attending the hackathon event after a rigorous social media campaign (i.e. Twitter, school mailing lists) by the event organizers. To curb the limitations of self-selection, which may include a decrease in engagement resulting from sharing similar interests (or the contrary), I attended hackathons in several industries (e.g. journalism, government, healthcare and education) to create more differences in the networks of participants.

The demographics were computed for four hackathons in my sample (two in journalism, one civic and the last one in healthcare). The first column in Table 2 includes results from a survey conducted by Major League Hacking on 1,339 student hackathon participants. Except for one

hackathon for which the organizers intentionally calibrated the female-to-male ratio to be 50:50, the hackathons in my sample only show a little more diversity than in the MLH survey, instead staying predominantly White and Asian with only one-third women. Most hackathons were developer-centric. The same journalism hackathon to reach gender parity also included more designers and people with product experience, and the healthcare hackathons invited more mechanical engineers. These demographics reflect the superiority of one group of workers over degraded labor.

Table 2 – Demographics for four hackathons

Event Name	Major League Hacking	Hacking iCorruption	2014 MIT Grand Hack	Hacking Journalism Boston	Hacking Journalism NYC
Event size	920	63	500	61*	85**
Application Required? Y/N	N/C	No	Yes	No	Yes

Gender	MLH	HiC	GH	HJ1	HJ2
Female	15%	25.4%	35%	29%	49%
Male	85%	74.6%	65%	71%	51%

Race	MLH	HiC	GH	HJ1	HJ2
White	45%	63.5%	N/C	69%	66.4%
East Asian	44%	33.3%	N/C	29%	25.6%
Hispanic	7%	6.3%	N/C	2%	6.4%
Black	3%	0%	N/C	0%	2.6%

Technical Expertise	MLH	HiC	GH	HJ1	HJ2
Engineers	20%	34.9%	29%	N/A	N/A
Designers	N/C	6.3%	7%	17.8%	61%
Developers/CS	60%	27%	13%	46.6%	24.8%
Scientist	N/C	7.9%	13%	N/A	N/A

Other Professions	MLH	HiC	GH	HJ1	HJ2
Clinician	N/C	0%	16%	N/A	N/A
Business	N/C	15.9%	18%	6%	41.9%
Journalists	N/C	6.3%	N/A	29%	43%
Policymakers	N/C	12.7%	N/A	N/A	N/A
Educators	N/C	1.6%	N/A	N/A	13.3%
Videographers	N/C	0%	N/A	N/A	32.4%
Students	N/C	28.60%	39%	N/C	7.60%

*Demographic information only available to 45 participants.
**Demographic information only available to 78 participants.
N/C: Data not collected
N/A: Non-applicable

Out of 100 participants, the following observations were logged, resulting in 45 transcripts so far, but of which only 20 examples and only 8 teams were used in my analysis.

Table 3 – List of Examples in this Dissertation

Team	Technology	Type of Play Labor by Example	Speakers
Campaign Con	Audit	Chapter 6; Ex 1-6 Hacker Literacies	5 out of 6
LIBOR Alt + Repair	Alternative to Markets	Chapter 5; Ex 1-4 Simulation Game	4 out of 6
Townhall	Platform	Chapter 2; Ex 1-3 Humor & Hobbies	4 out of 4 (FULL)
Sirona Care	Wearable	Chapter 3; Ex 1 Distributed Cognition Chapter 5; Ex 2 Paired programming as Disciplined Play	6 out of 9 (FULL)
Reciprocall	Telehealth	Chapter 3; Ex 2-3 Role Reversal Ex 4 – Where's the doctor?	3 out of 9
The Cuff Guys	Portable Hardware	Chapter 3 Pivoting	4 out of 4 (FULL)
Newstritious	Recommendation Systems	Chapter 4; Ex 1 Technical Mis-translation (language-specific)	6 out of 6 (FULL)
Video Pizza	Converting Video to Text	Chapter 4; Ex 2 Non-linearity	9 out of 9 (FULL)
Total teams: 8	**Preferences for Tech** Journalism: Machine	20 examples total	41 spoke out 53 speakers

	Learning & Video Healthcare: Hardware & Telehealth Government: Platforms and Audit Tools		

Limitations of the study

The biggest limitation was my inability to personally watch the code that each participant wrote at the event. Even in the advent of filming, close-ups of the video or enlarging the video in the video could not show the content of their screens clearly. Not having access to it in real-time, I could not embed the code in the team's interactions. Neither could I keep a log of all the resources they consulted to identify which in particular helped them learn, and distinguish what they wrote from what they copied and pasted from libraries. I therefore am not able to comment on the coders' learning style; only when they delegated could I know who was an expert and how they approach a problem. Fluctuations in their activity levels can also be a proxy for their learning curve. Otherwise, an advanced programmer can tell whether the author of a code is experienced by how compact, elegant the code is, and how cleverly it is laid out.

Often at the event, participants in my study would enroll out of curiosity for my evaluation. Being positioned as an expert was enabling in a way, I was able to recruit more easily. Most teams were too busy to be influenced by my presence, and they let me quietly record them. For the few overly enthusiastic ones, I had to explain that I needed time to produce an analysis and gave them my card. Some teams directly worked me into the audio recordings, but I made sure to leave the room periodically to produce undisrupted transcripts. The reception of these intrusive methods ranged widely, with one journalist encouraging me to position my recording equipment – one of two smartphones – where I could hear them better, to the shy developer shifting locations to avoid

being recorded. As I did not have much time to build rapport with any participant prior to the event, and was often too busy getting all the informed consent forms signed, I did have access to conversations that were not about work, and often by sheer luck, caught the team in interesting moments.

Having known me from their first hackathon, the Hacking Journalism NYC organizers were familiar with my research. Handing me fancy video equipment, they charged me with filming their event and had me interview one organizer and one participant for a press release they were putting together. On the organizer side, I often had to do work to gain a stronger foothold in their network, probably due to liability issues. In exchange for a blog entry on mentorship models at a hackathon, I was eventually able to get my hands on scanned copies of the judge's comments for Hacking iCorruption, albeit at a relatively late stage of my fieldwork. After consulting with one of the organizers from Hacking Medicine, I received the tally sheets for the Telehealth teams that participated in the 2015 Grand Hack. Sometimes, I followed participants after the hackathon to a bar (Meadhall in Cambridge) where I was able to observe a sponsor mingling with a team. Most often than not, I was not allowed in meeting rooms where sponsors might be, and most of the research on these companies was in fact archival.

From studying one-shot events, an irreconcilable drawback was participant attrition from the follow-up survey. Some were bad at responding to, or maintaining contact over at e-mail, while other ones are out of state. I developed the following two approaches to tracking my informants: 1) on the one hand, I developed rapport with Ying and Jihii from the Newstritious team; Tarek, Andrew and Khalil – who later became an organizer for Hacking Medicine – from The Cuff Guys team; Aaron and Fredrick from the Anti-Corruption University team; and Max from the CampaignCon team, whom I followed as they participated at other hackathons after I recruited

them. I was also able to log repeat visits to the civic organizations that participated at Hack 4 Democracy during their weekly meetings in New Hampshire, and even met Professor Lessig at a private fundraiser there. The benefit of following a person over an extended period of time was that I could cross-reference what one says in different situations, over a same subject and check my own biases. I developed a relationship with organizers Daniel Miller and Matt Caroll behind Hacking iCorruption, and Priya Garg at Hacking Medicine, who had been assigned to be my first point of contact for the planning of the Grand Hack and whom invited me to informally sit at multiple organizer meetings.

My second, relatively weaker approach was to keep track of projects. I checked for updates to code repositories, but developments had to be cross-referenced by leaders of the projects to make any sense, given that the author listed on a page sometimes was not the original author of the code in the transcript, and had simply uploaded the file on behalf of everyone.

Timeline

On-site observations and recruitment took place between June 2014 and May 2015; in-between events, I attended many meetings and conducted follow-ups, as well as archival research, coding and analysis.

June 2014: Hacking Journalism Boston (test run)

With the help of Graham Jones's graduate research assistant, I was able to get video footage of team formation for the first hack in my sample and annotated transcriptions of two teams. I was able to conduct follow-up interviews for most members of both teams we observed. For this test run, we spent the summer and fall studying team interactions from several angles before I went back to the field in September.

September-December 2014: Hack 4 Democracy, Athena Health MDP's first hackathon, Open

edX hackathon

After attending the first civic hackathon, I followed two leaders of some of the organizations who had been present to their headquarters in New Hampshire. I could then see whether they had implemented the hacks post-hackathon, and how they incorporated technology in their workflow. The only hack to be adopted, I learned, was Anti-Corruption University. ACU launched in January 2015. I followed up on its reception with San Francisco-based coder in February 2015 while I was visiting, but it was unsuccessful.

October was dedicated to these meetings and following up with two teams from the civic hackathon in late September. November saw too many healthcare hackathons to follow. I met the MIT co-directors at the demos for one, who directed me to a company hackathon they had helped advertise with their name while I waited to see theirs in April. One week after the Athena Health hackathon, I helped a fellow anthropologist film the Open edX hackathon. Since Shreeharsh was the expert on Mass Open Online Courses (MOOC) and knew the daily work of these educational nonprofits, I moved on to attend three healthcare hackathons instead.

With the help of Shreeharsh Kelkar, I obtained footage of one entire team with complete transcriptions on-site. Shreeharsh was kind to help me follow up with members of the team he observed.

January 2015: Student Hackathon Organizer Conference and Hacking Journalism 2 in NYC

I spent the blizzard month of January and the early part of February 2015 at home in Brooklyn. A participant at the first journalism hackathon forwarded the application to a second journalism hackathon organized by the same group of people. Two weeks after attending this one, I attended the second annual conference on hackathon organizing. My focus was more on the organizers at this time. I learned how they came together.

March-May 2015: Hacking iCorruption, MIT Grand Hack

By late February, I had received a phone call to discuss my plans to attend the weekly organizer's meetings for the MIT Grand Hack in April. I spent two months recording observations. In-between, a participant from the first journalism hackathon invited me to research the second civic hackathon, whose lead organizer I already knew from the first one.

Summer: Re-read on information theory; organize data

In anticipation of Chapter 3 on value, I spent the summer revisiting information theory with the help of a civic hackathon participant. I had concluded my two semesters of observation, and started to organize my data and transcribe.

September-November 2015: Follow-up interviews, surveys

I researched the professional backgrounds of all the participants at four hackathons, and conducted my last follow-up interviews.

LITERATURE REVIEW

In this review of the anthropological literature on play, I fuse Durkheimian and Marxist traditions and highlight the analytical distinctions between play and work that I have found helpful in my framing the hackathon as a site of cultural production.

If one could look at the hackathon model as a potlatch, a tech fest in which a number of host institutions pool and distribute gifts ("swag") according to each guest-worker's rank or status, one finds some important commonalities between these finite events and Mauss's politics of the gift, which de Certeau (1984) acknowledged as a diversionary tactic (27) and Boon (1999) later reprised in its evocation of an excess of ludic expenditure (160). What Appadurai (1986) does differently is embed its leisurely and simultaneously competitive connotations in economic relations. Re-examining the potlatch, Appadurai frames these complex rituals as tournaments of value, which he defines as:

> "Complex periodic events that are removed in some culturally well-defined way from the routines of economic life ... what is at issue in such tournaments is not just status, rank, fame, or reputation of actors, but the disposition of the central tokens of value in the society in question. Though such tournaments of value occur in special times and places, their forms and outcomes are always consequential for the more mundane realities of power and value in ordinary life. In such tournaments of value, strategic skill is culturally measured by the success with which actors attempt diversions and subversions of culturally conventionalized paths for the flow of things. (Appadurai 1986, 21)

I take from Appadurai's framework the following observations:

"*…what is at issue in such tournaments is not just status, rank, fame, or reputation of actors, but the disposition of the central tokens of value in the society in question*." At a hackathon, status is achieved in this mode of competitive giving, precisely in those disciplinary moments that propel participants to question their core values.

"*In such tournaments of value, strategic skill is culturally measured by the success with which actors attempt diversions and subversions of culturally conventionalized paths for the flow*

of things...” Despite its Marxist overtones, as Appdurai’s notes are on competitive giving serve to remind in the potlatch certain potential principles of inalienability that drive the social production of artifacts and relations, which do not have anything to do with monetary value. The tournament of value is sufficiently removed from ordinary life and real commodity-exchanges to facilitate this principle. However, very much like Malinowski observed in the *kula* exchange later, for which men would risk life and limb to travel across huge expanses of dangerous ocean to give away what appeared to be worthless trinkets, we must ask why these hackers would so readily give up their weekend to build something that did not directly fulfill an economic purpose, and why companies are willing to “pay to play” if they cannot play.

“*Though such tournaments of value occur in special times and places, their forms and outcomes are always consequential for the more mundane realities of power and value in ordinary life*.” We must also ask how displays of technical prowess, which make some hackathons highly competitive events, relate to forms of political authority – to prestige – in this potential potlatch of skills, technology-objects and labor. Whether, indeed, these diversions are *consequential* and lead to a deferred wage, or rentable labor to work on side projects after the hackathon.

Sutton-Smith’s (1997) comprehensive typology of play under seven rhetorics helps us recognize the full gamut of play types that are at stake at a hackathon. Among the scholars he lists under the seven rhetorics in Table 4, Huizinga (1938, 13) was the first sociologist to define play: “a free activity standing quite consciously outside ordinary life as being not serious,” absorbing the player intensely and utterly but not connected to any material interest.

Table 4 – Sutton-Smith's Typology of Play

The Seven Rhetorics of Play

Rhetoric	History	Function	Form	Players	Discipline	Scholars
1. Progress	Enlightenment, evolution	Adaptation, growth, socialization	Play, games	Juveniles	Biology, psychology, education	Vygotsky, Erikson, Piaget, Berlyne
2. Fate	Animism, divination	Magic, luck	Chance	Gamblers	Math	Bergler, Fuller, Abt
3. Power	Politics, war	Status, victory	Skill, strategy, deep play	Athletes	Sociology, history	Spariosu, Huizinga, Scott, Von Neumann
4. Identity	Tradition	Communitas, cooperation	Festivals, parades, parties, new games	Folk	Anthropology, folklore	Turner, Falassi, De Koven, Abrahams
5. Imaginary	Romanticism	Creativity, flexibility	Fantasy, tropes	Actors	Art and literature	Bakhtin, Fagen, Bateson
6. Self	Individualism	Peak experience	Leisure, solitary, extreme games	Avant-garde, solitary players	Psychiatry	Csikszentmihalyi
7. Frivolity	Work ethic	Inversion, playfulness	Nonsense	Tricksters, comedians, jesters	Pop culture	Welsford, Stewart, Cox

Source: Sutton-Smith 1997, 13.

Huizinga (1955) is classified under games whose form – skill, strategy and deep play, to which I return shortly – produce a rhetoric of power; Piaget (1945) and Vygotsky (1933) follow a rhetoric of progress; Bateson (1972) and Bakhtin (1941) appeal to the imaginary, and Turner (1982) falls under play as identity. Sutton-Smith classified contest play under rhetorics of power, which is a slightly useful distinction given that hackathons recognize individual merit but do not fix statuses thereafter. In chapter 5 of *Ambiguity of Play*, however, Sutton-Smith (1997) demonstrates how these categories of play are not mutually exclusive. So if power-play, contestants are said to get immense satisfaction from specific actions outside of themselves and specific to the game itself, oftentimes contest-play is also a play of disorder, such as carnival, and deep play, at which point

Sutton-Smith enters into a full discussion of Geertz's (1972) proposition that contest-play serves as a text to interpret power relationships. We see in the collective actions taken by participants at hackathons that power interests are intrinsic in contest-play, such as the application of skill and leadership in actions and strategies e.g. who gets to be leader, who gets the lowest roles, who gets eliminated, making power an essential ludic concept. Their tactics of ambiguity, very much like the cockfight, help them experience variable success in seeking recognition for their hard work.

Studies of play gained momentum with the rise of educational video games (Gee 2014; Hung 2011; Ito 2009; Salen 2008), corresponding to the development of constructivist pedagogies to teach children programming and engineering concepts through games (Brennan and Resnick 2013; Kafai and Burke 2014; Santo 2012). At higher stakes competitions like the NYC Big Apps, which is sponsored by New York Economic Development Center and offers $10,000 to $30,000 prizes, participants are highly encouraged to play as well, and are given play-doh and thread (Figure 2).

Figure 2 – Brainstorming with Play-doh
Source: NYC Big Apps Prototyping Workshop, March 22, 2017.

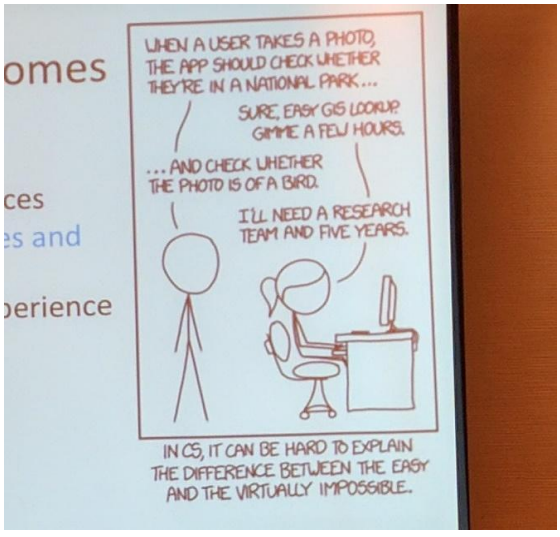

Figure 3 – *Lulz as* Part of the Coder's Life
Source: CCSW Hackathon Research Group Workshop, February 25, 2017.

Yet, despite the pedagogical and economic turn to play, Sutton-Smith (1997) reminds us that different play-types point to very different ideologies. The past decade has seen a boom in scholarship on the role of play in boosting productivity and creativity inside companies (Berg et al.1995; Schrage 1999). The expression of this ability to think outside the box and create something is, for one, informed by constructivist theories of play, which draw both rhetorics of the imaginary and progress. On the other hand, workplace play – the beginnings of play labor – is better characterized by a cartoon[8] (Figure 3), in which a programmer radically changes his statement once his boss adds a specification. In guise of a subtitle, the following statement was injected: "In computer science, it is hard to explain the difference between the easy and the virtually impossible." Gabriella Coleman's (2012) *Coding Freedom* would support this statement. Coleman shows that the politics of Free Software cannot be separated from the vagaries of everyday production and animates the difference between what could be fun and painstaking.

Play does not preclude serious political tactics to curb authority, and its visual spectacle brims on the edge of risk. In the case of Anonymous, we see that hacker's play can transcend any political imaginary and result in real-world security breaches. Lilly Nguyen (2016) even goes deeper: in her study of hacking iPhones in Vietnam, this transgressive practice is intended to break a particular techno-culture – its laws and related work practices. In his study of the infrastructure of piracy in Northern Nigeria, Brian Larkin (2008) describes how routine repairs incur its own cultural experience of technology (233, 241). This literature on hacking does not preclude Geertz (1973) and Ortner's (2001) earlier interest in deep play as high-stakes rituals, which reproduce the culture in which they are enacted. In the same tradition of social action, Jill

[8] In the world of geeks and techies, this kind of cartoon humor is called "lulz," and is a form of political humor (Coleman 2011).

Koyama and Herve Varenne (2012) documented the making of No Child Left Behind in NYC after-school programs, tracing the formation of actor-networks in their implementation under circumstances of failure in which these very groups are singularly caught. I am not so interested in deep play as I am in what this line of research draws from de Certeau's (1984, 36) notion of "making do," brilliantly described as the manipulation of imposed spaces or tactics in a certain art of diversion. In the absence of a proper locus, play "takes chances on the offerings of the moment." I am also not as interested in how actors collude and orchestrate failure (McDermott and Varenne 1995), or even how leisure and work have been sanctioned, but I am interested in what people do to curtail actual material constraints of the machine, and build from this broad discussion on play to study this kind of labor.

Another important contribution, Bateson (1972, 317), established that play was a form of metacommunication, through which a speaker's utterances implicate that he is not serious. It was in front of his daughter Catherine, who insisted that he be truthful, that Bateson formulated his first thoughts on play: as of relatively low consequence for those involved. On that count, Suchman (2007) seems perplexed, buttressing the Batesonian thesis that participation in a game can be accounted by a player's understanding of the rules, arguing that a situation of technological breakdown could threaten that logic any time. Preferring to side with Garfinkel (2006), whose notion that one eventually finds himself playing and devising new orders is central to her work on artificial intelligence, Suchman shows how the limits of a design do not simply result in a misunderstanding. Instead, participants are seen to engage in a wide range of creative appropriations – an assertion of de Certeau's (1984) *language parlant* of the algorithm. That being said, play can also be risky (Geertz 1992), even extravagant in the face of ambiguity (Boon 1999).

I wish to go beyond the language of deep play (Geertz 1972) and subversion now, to address some of the pertinent issues and forms of play labor that are central to the school of digital labor. Play labor (Terranova 2000, 32) has its origin in digital labor; it borrows its name from leisurely forms of free labor on the Internet that are embedded in capitalist flows. Digital labor encompasses participatory culture such as fandom, video games, and social networking; it also involves algorithmic labor and the activity of hacking as a form of cybernetic control. I broaden the definition to include, among other things, the social production of new tools to power the growth of digital labor and the classes of workers that produce them. Such work of the latter kind sometimes requires the coordination of digital labor, which I argue is supported by play labor and is the premise of a hackathon: setting aside 48 hours in the ordinary lives of digital workers, these complex episodes draw on speed, flexibility and mass participation to engage communities of practice in collectively reflecting and exploring new modes of value capture on the Internet, while at the same time, using technology as a prop to better align with their industry contexts.

Digital labor is first and foremost the process of extracting value from the means of communication. But there is more to its relationship to the commodity. In *Labour and Language*, Virno (2001) writes:

> Living labour, an appendix of the system of machines, follows a natural causality in order to use its power: what Hegel called 'cunning' of labouring. And 'cunning' is known to be taciturn. In the postfordist metropolis, on the other hand, the material labouring process can be empirically described as a complex of linguistic acts, a sequence of assertions, a symbolic interaction. This is partly due to the fact that now labour activity is performed *aside* the system of machines, with regulating, surveillance and coordinating duties; but also because the productive process uses knowledge, information, culture and social relations as its 'primary matter'.

'*… a complex of linguistic acts, a sequence of assertions, a symbolic interaction'*: Using the raw

materials at his disposition, namely his language[9], the new worker-subject shares his knowledge and generates information on his person and his interlocutors. Lazzarato (1996, 136) defines a kind of *immaterial* labor as "the expression of the new politico-subjective position of workers as producers of the informational and cultural content of the commodity." In anthropology, all action is a form of communication; it is the labor of achieving mutual recognition alongside machines (Suchman 2007) and it is that which produces a sign-relation (Saussure1916). The term 'immaterial' for digital labor is nevertheless misleading in light of recent scholarship on the materiality of new media. Lazzarato explicitly draws from Virno's (2001) argument that social relations have been dematerialized due to the noise of communication's labors and the characteristic of eminence in language. Digital labor is called immaterial because the processes of expropriation are invisible to the eye of users, but as Brunton and Coleman (2014, 78-80) suggest, the translation of the digital into analog means that the input-output infrastructure of the web does not in any way resemble language. Leave it to Tufecki (2016) to add that by design, the latest developments in machine learning, which now manage all of our business transactions, generate data separate from its designers, and therefore alienate some labor processes from human labor. Since programmers cannot even understand what their own program is doing most of the time, the work of humans would now consist of developing ways to learn how to learn from these intelligent systems.

In this regard, hackathons are fertile ground for studying the management of digital labor as a sequence of linguistic acts showcasing the process of learning how to learn through

[9] Saussurean linguistics treats language as matter. Producing meaning (the signifier-signified relationship) involves establishing a distinction between sounds [raw matter] in relation to one another. Virno (2001) recommends reading Saussure and other philosophers of language such as Wittgenstein and Carnap to understand the laboring praxis of 'the talkative factory." Value-adding labor is the activity and process of communication.

playbour. We find in Terranova's (2000) notes on the modding[10] phenomenon, some commonalities with its pleasurable aspects of communication and exchange in the hackathon. Because it is not imposed, play labor cannot seek compensation. Ross (2009) expounds the logic of play labor: "all of the interactive free labor that goes into user-generated value can be seen as kind of a tithe or tribute we pay to the Internet as a whole so that the expropriators stay away from the part of it we really cherish" (De Kosnick 2013, 107). The hackathon would therefore set the terms under which laborers are temporarily shielded from processes of extraction.

What we know of hackathons is that they usually take place on weekends during one's free time. Prizes are symbolic; some hackers might sacrifice sleep in order to win a gadget that they think "is nice to have," but the opportunity to freely exchange skills, learn from one another, or network with companies are other reasons to participate. Many ideas developed at hackathons remain in the initial prototype stage, if one can call a barely functional app a 'prototype.' As shared creative expression, hackathon activities stand to raise important analytical distinctions in other forms of digital labor.

First, play labor establishes the superiority of a group over another or vindicate a community identity through skill-based competitions. Niels van Doorn (2017) writes about how a shrinking middle class workforce has welcome new ways to market its assets, even if it is only their labor-power. This class of workers exhibits characteristics of superfluidity by the obfuscation of their identities, remote control and non-accountability outside of the commercial relationship they engage with a potential employer. Unlike these low-skill, highly racialized and feminized freelancers in van Doorn's account, hackathon hackers triumph over degraded labor by sharing the characteristics of being very mutable and rapid.

[10] Modding is a slang expression that is derived from the verb "modify". Modding refers to the act of modifying hardware, software, or virtually anything else, to perform a function not originally conceived or intended by the designer, or achieve a bespoke specification.

And yet, play labor cannot be separated from the reverberations of crises in the cultural sphere. I am only talking about on the hand, a crisis of profit, whereby the labor of maintaining a free medium can only exist because of venture capital, and on the other, the crisis in the cultural authority of certain practices of knowledge production. The framework of tournaments of value brings to the study of play labor a deeper understanding of how workers use the hackathon model to introspect into their own ways of working. At the same time, we benefit from inquiring into how the modes of participation from designers, developers and industry professionals highlight the more local experiences of new networking technologies and fuel the growth of digital labor. Traditional conversation analyses help us see how they use these experiences to fabricate a context for these technologies and re-appropriate work as their own, as well as how they engage and manage their collective intelligence – a paramount issue in digital labor that goes hand in hand with the crises of profit and authority.

Having fully discussed play and its relation to digital labor, its potential relationship to deep play, transgression, strategy and the imaginary, I now move on to a full general description of how playbour is generally organized at a hackathon (Chapter 1).

Chapter 1: The Who, What and Why of Hackathons

Organizing hackathons takes up anywhere between one and six months. Based on seven hackathons I observed in three industries, I describe the actor-networks that make up three types of hackathons – the physical space, who sponsors what, who advertises the event and where, who chooses the venue and pays for it, and who serves as supporting staff on the day of the event. The different models we see all borrow central tenets from a certain hacker ethic to manage the ludic aspects of the event. Hackathons try to remove participants as much from everyday life as possible, but they also happen to be sponsored by companies looking to learn from their participation. It is therefore play from sandboxing production, but participation is possible from exploiting forms of cognitive capitalism.

Modeling hackathons

There is very little in the way of details to describe what a hackathon looks like. Two terms resonate from any invitation to participate at such events: "like-minded peers," driven by passion for a subject and "innovation" – creativity. A Facebook organizer told me that one only needs two things to make a hackathon happen: enough people and lots of ideas (Thomas Lai, pers. comm. March 3, 2017). The invitation to join like-minded peers paints a simple, albeit captivating, picture of what hackathons are: collaborative production events, where teams self-organize around interests, skills and their passion for a technical project. Initially, I wondered just how different these were from other types of meetups – with comparatively low barriers to entry and similar ritual spaces that let anyone pitch an idea. The difference was in the level of commitment: hackathons explored *new* problems that arise every year from a new field of technology, whereas meetups held talks and workshops for those motivated to contribute to *ongoing* projects. Most hacks from hackathons do not become full-fledged open-source projects. Using the discourse of "innovation," hackathons drew on elements of spectacle to attract more

diverse actor-networks at events. The event started with pitches – proposing a solution that had never been built before – and ended with demos of a prototype, also termed a "proof-of-concept" (Jacob Schloss, Startup Weekend communication, March 29, 2014). In some cases a business proposal is traded in, to get participants thinking about how their designs could affect the world outside the room. Together, the spectacle, the fanbase being cultivated around new media technologies, its focus on building and breaking, and affinity-based networking make up for the fun part of the weekend.

This chapter is therefore devoted to the kinds of engagement a hackathon produces that are paradoxically free, creative and functionally engaging. Questions around increasing diversity, timing events, and how to reward volunteers, are some of the more obvious and most problematic structural dimensions in its organization. I describe how participation at hackathons is a form of play labor, fulfilling a demand for flexibility, on-demand labor in the digital economy. One only needed to look at the value proposition that was being made to sponsors of the MIT Grand Hack to understand why companies were interested in sponsoring a hackathon:

1. A novel model with which to approach the innovation process. Many of our sponsors are inspired to conduct internal hackathons after working with us.
2. An energizing environment that brings people together and acts as a catalyst for networking and recruiting. Hackathons are excellent recruitment events because companies get to see participants in action and working in teams.
3. Opportunities to be showcased in the case of media attention around the hackathon.
4. A first glimpse at novel projects that effectively address a variety of healthcare problems.

Figure 4 – Value Proposition to Sponsors
Source: MIT Hacking Medicine Sponsorship E-mail Template.

This letter to sponsors lists four features of hackathons as compelling motives to companies looking to draw attention to themselves and network with their future employees. First, hackathons could advertise the company and give its representative or recruiter a head start in recruiting. As such, there is value in seeing teams "at work." Additionally, leaders in industry

could get a bird's eye-view of the internal workflow of engineering teams and come to normalize creative projects. It therefore did not come as a surprise when this technology became commercially viable the following year. The press had started to promote them by then, and the next hackathon would continue this work, making novel approaches and applications into a topic of tremendous practical interest. Works-in-progress with the potential to solve a current problem in industry drew onsite the very people with the resources and talent to see these projects through. By pooling this retroactive labor at the event, it was easier to generate the feedback startups, companies, nonprofits and schools desperately needed to think through their technology projects.

While hackathons drew in a certain number of freelancers looking for work, as well as students hoping to win an internship, these motivations seldom accounted for how engaged people felt once they entered that space – its "energy." In moments of collective effervescence (Durkheim 2001, xx), when everyone was sprinting to finish, or enjoying networking and exchanging ideas, these freelance workers would not consider themselves "at work." Participation was optional and open to the public. Because this voluntary, ludic participation removed participants from the precarious, contractual conditions of freelance labor. Participants did not receive any wages for participating, but received, in exchange for their free labor, all the benefits of having scarce resources at their disposal. When tied to corporate intentions, a poorly designed hackathon could make the participant feel that he was being exploited in exchange for food. Ironically, the most adept hackers expected to eat well (Dan Muldrew, interview, May 24, 2017). As Lawrence Lessig explained a month after his staff hosted Hacking iCorruption, "something magical happens when you put people in the same space with food and drink" (speech, May 1, 2015). Beer was not a means to quench one's thirst, but its ritual efficacy at hackathons could be demonstrated by the networks of people that could made to cohere from its

consumption; it loosened strangers and was responsible for the collective hive effect and the free flow of ideas.

Hackathon participation, despite being tied to corporate sponsorship, had to be allowed to reside outside capitalist production to engage the diversity of event participants. Scholars from the "Hacking as Time-Mediated Events" Workshop at the 2017 Computer-Supported Cooperative Work Conference distinguished between events that were instigated by the community, and events by external partners meant to engage. Drouhard et al. (2016) identified three models for organizing participation at data science hackathons (Table 5).

Table 5 – Typology of Hackathons

TABLE 1
Typology of Hackathon Events

Keyword	Communal	Contributive	Catalytic
Primary Purpose	Developing resources, infrastructure, practice, and culture for community	Contributing to larger effort by focusing on discrete, modular tasks	Demonstrating utility of dataset, technology, or approach
Challenge	Advancing methodological approaches in the community	Completing as much work as possible in a fixed period of time	Generating an idea both novel and tractable
Seed	Specialized methods	Defined task	Articulated challenge
Motivations for participation	Professional development	Impact	Recognition
Mode of participation	Collaborate	Execute	Innovate
Style of Work Environment	Didactic	Autonomous	Competitive
Qualification for participation	Membership in community	Concern for the project/issue	Skills or an idea
Continuity	Of community	Of project	Of ideas

Source: Drouhard, Tanweer and Fiore-Gartland 2017, 3.

These three models were a starting point for a heated discussion on what the purposes and outcomes of a hackathon should be. The MIT Grand Hack, for instance, clearly fell under the category of "catalytic" hackathons, with the primary purpose to explore novel approaches, provide the seeds to articulate a challenge, and support motivated participants in gaining recognition for an innovation. Civic hackathons could qualify as contributive, with assigned volunteer tasks and focused more on impact than creativity. More communal types (i.e. journalism) hackathons brought in journalists to participate in professional development by way

of collaboration between news agencies. Such modes of participation invited different modes of engagement and entailed measuring success differently, including but not limited to: growing a community of practice (Lave and Wenger 1991), cultivating leadership in open-source projects, and ideas gaining popularity.

The historical formation of hackathons

The history behind each hackathon provides some important insights into the types of labor to be found in this dissertation, from forms of political action to play labor that did end up producing startups, to training for entrepreneurial subjects. Hackathons are actually a fairly recent affair. The first appearance of hackers in popular culture dates from the 1970s, but hackathons first started in international development. Since then, their applications in software and hardware development have multiplied.

Inspired by the steady rise in technical volunteerism in response to global humanitarian crises, the World Bank started hosting hackathons before the commercial sectors caught on, mainly to raise awareness of water-related challenges and create novel applications to improve infrastructure in resource-constrained areas. Though the World Bank likes to credit LinkedIn and Facebook for popularizing hackathons (Bank 2012, 2), civic hackathons were organized in the budding e-government sector as early as 2008. The 2008 Apps for Democracy is the first occurrence Johnson and Robinson (2014, 351) report, sponsored by the city of Washington D.C. to mark the creation of a municipal open data catalog. With open data on the rise, an alternative to the procurement process then existed for outsourcing government technology or software, without direct oversight by the government except in the provision of data.

By the time Chris Kelty (2009) came up with the notion of recursive publics, the functions of a hackathon had already changed to keep up with the demands of venture capital for enterprise innovation. Despite its current mainstream associations, I argue that recursive publics

make the hackathons what they are: microcosms of industry. Chris Kelty (2008, 1) defines a recursive public as: "a public that is vitally concerned with the material and practical maintenance and modification of the technical, legal, practical, and conceptual means of its own existence as a public; it is a collective independent of other forms of constituted power and is capable of speaking to existing forms of power through the production of actually existing alternatives". At its most basic, a hackathon existed to promote the forms of digital labor that supported its participants' livelihoods. As "*independent of other forms of constituted power*" as hackathons could be by removing participants from everyday life, its participants formed recursive public by the manner in which they assembled: in the grapevine, an e-mail announcement on a listserv, or a tweet by a person they followed on social media. These people convened at a hackathon because they were equally concerned with the practical maintenance of the technical infrastructure on which they did work. Their association with certain companies may temporarily make them what Adam Fish (2014) calls an "organized public," one of the many recursive publics on the Internet, which might stand in relation to a former social enterprise but for which membership or other forms of belonging are temporary and informal, like a listserv. The total ensemble of companies, judges, organizers, journalists, developers, APIs and code repositories made one hackathon, one actor-network (Latour 2005). One hackathon was one actor-network made up of historically loose affiliations.

Hackathons work as platforms for valorizing people through playful negotiations of the means of production. In Chapter 6, the case of Campaign Con demonstrated that strangers were quick to establish each other as "gurus" or amateurs. It goes without saying then, that we can find the full spectrum of talent at a hackathon, and one would want to be there because important people from industry will be there. As no anthropology of the disciplinary worker-subject is complete without a list of those who hold power at these events, I now list the sponsors and

organizations that delineate the boundaries of the digital economy for journalism, civic technology and e-government, as well as healthcare.

Hacking Journalism NYC was organized by Hacks/Hackers, a data journalism education nonprofit started in 2009 under the auspices of the Knight Foundation, in partnership with the MIT Media Lab and Embed.ly. The Hacking Journalism took place in a different city every time and aimed to build the community of dispersed technologists all over the world interested in advancing new techniques of data journalism. The goal of the journalism hackathons was to identify the next trends in journalism, and provide an opportunity to data journalists to pick up new skills. In 2014 and 2015, the themes were set by the choice of media: mobile, video and data science. Participation stopped when the core group of organizers stopped organizing these events, shortly after the July 2015 hackathon in DC.

Regardless of who the sponsors were at each event, it was not uncommon to see the Knight Foundation listed for any related data journalism activity. Most data journalists passed through one of its fellowship programs, the reputed pipeline for placing computer scientists in newsroom internships; in response to the growing demand for their skillset developers in the newsroom, most data journalists producing the algorithms writing the news and data visualizations were for in fact not traditionally-trained journalists, but developers who apprenticed in a specialized set of methods for producing the data for the news.

It should be noted that most hackathon organizers did not commit to host these events indefinitely, and often renounced their roles within the year in search for more lucrative opportunities to grow their resumes. Matt Carroll eventually transitioned out of his role as researcher at the MIT Media Lab, after which time the FOLD project he had advised stopped being updated. Jeanne Brooks, then the first-ever executive director of Hacks/Hackers, grew out of her vagabond lifestyle that had bonded us at our first meeting. She is now a senior advisor to

the Local News Lab at the Dodge Foundation in DC, but unlike Carroll, she had continued part of her freelancing, itinerant journey, organizing Emojicon – another "first-ever" in her list of achievements – and a celebration of all things emoji in San Francisco on November 4-6, 2016.

Perhaps the lack of an infrastructure for updating data journalists in the newest techniques for making sense of datasets was the real source of frustration for *The New York Times* in 2017. The one-shot, hands-on professional development opportunity was nevertheless most welcome in 2014 and 2015 in the absence of any certification for keeping up with the pace of innovation. In mid-2014 when I started fieldwork, Hacking Journalism teams were just figuring out the script to automate the selection of the most important content in a written article. Having lost advertising revenue to Facebook, news producers looked to video and augmented reality to offer an immersive experience that engaged readers' affective response to the news. More readers were consuming news on their smartphones, and virtual reality was predicted to soon reach the hands of consumers after. News agencies such as *Al Jazeera* and *the New York Times* were preparing to make the transition by first seeing how their staff could first learn to streamline video news. For them, Hacking Journalism was just another inter-agency potlatch. Hacks/Hackers served as an intermediary event planner, organizing the journalists around free workshops it offered.

Hacking iCorruption slightly differed in the structure of its actor-networks, which served not a particular class of activists, but the specific needs of the Harvard Center for Ethics. However, its sister hackathon Hack for Democracy also catered to the needs of organizations in the New England area, where on many boards its director – celebrity law professor Lawrence Lessig – sat. Brooke Williams, a fellow at the center, had pitched the idea of holding a hackathon. In her follow-up interview, the Pulitzer-winning journalist explained that for a long time, she had wanted to build an online database to share her research on foreign donations to

think tanks, but did not have the resources to secure the commitment of a developer to do it. Initial programming for Hacking iCorruption further came from Maggie McKinley, a lawyer and lecturer at the Harvard Law School; the event borrowed her design for the higher-stakes civic hackathon series at Harvard that same season – Hack for Congress. While Ralph from the Townhall team in Chapter 3 had suggested that Daniel Miller was a bigger node than Lawrence Lessig was, the actor-network for the Hack 4 Congress series that ran parallel to Hack 4 Democracy Boston and Hacking iCorruption that same year and whose core allies are listsed in Table 6, indicate that Professor Lessig and Ms. Kinley used their connections to the Sunlight Foundation and the OpenGov Foundation, mostly as members of the Board of Directors and Advisory Board, to fund and organize different events in the series. One can see that Cambridge does not do small, with easily ten core allies per hackathon among the most influential or largest organizations in DC.

Table 6 – Actor-Networks behind Hack 4 Congress DC

Hackathon	Organizer	Sponsors	Connection	Outreach Through	Area of Focus
Harvard 4 Congress DC	Ash Center	Google		The Hill	
	OpenGov Foundation	Microsoft	Maggie McKinley is on the Board of Advisors	Advisory Committee on Transparency	Educate Congress of transparency issues
	Capitol Bells	Shuttleworth Funded (based in South Africa)	Fellowship; "Open" Philosophy of the Knowledge Society	Capitol Bells	App to notify of floor proceedings
		Knight Foundation (based in Miami)		Congressional Data Coalition	train Congress to respond electronically to citizens
		Democracy Fund (in DC)	Part of the Omidyar Group	Congressional Management	Train Congress to respond

			(founders of e-Bay)	Foundation (based in DC)	electronically to citizens
		Facebook		The Curious Citizen Project	Civic art
				Data Transparency Coalition	Non-proprietary data standards for the information they generate or collect, and publish such information as machine-readable data
				FWD.us	Immigration reform
				GovTrack.us	Track bills
				Millenial Action Project	Forge pragmatic cooperation on future-oriented challenges
				OpeningParliament .org	
				PopVox	Verifies, aggregates and simplifies communication with Congress on an open, trusted and nonpartisan common ground.
				Quorum, data-driven politics	CRM for legislative workflow and tracking
				R Street	policy think tank for free markets and limited, effective government
				Sunlight Foundation	

Source: Hack 4 Congress Website 2015.

Hack 4 Democracy sought to increase access to technical talent for civic organizations,

not to nurture it like the Hacking Journalism series did. Team Democracy was based in San Francisco, and was called upon to share its expertise in hackathon organizing. Represent.us, the Massachusetts nonprofit which Lessig served as a Board member, included Harvard student representative of MA-07 district and Democracy Matters student representative Jonah, MIT student representative Danny Miller and outside affiliate Dan Wong. All three young men helped organize the hackathon, with Dan Miller using his experience to organize Hacking iCorruption later the following year. Represent.us, the New Hampshire Rebellion and Democracy Matters were all activist organizations based in Massachusetts dedicated to fighting big money politics. Asking participants to draw how each organization was related to one another was a difficult task in light of a new merger between Open Democracy and the NH Rebellion, even prior. Lessig was the director of the Harvard Center for Ethics; the Safra fellowship was Lessig's biggest asset, pooling the brainpower of academics and developers with a commitment to researching institutional corruption. The civic hackathons I observed were organized exclusively to support Lessig's network of activists working fervently on his campaign.

Those who participated at Hack 4 Democracy Boston were therefore all connected or inspired by Professor Lessig, the mythical father of the campaign finance reform movement. Lessig drew a sizable audience to his pre-hackathon panels among those who believed in his mission and life's work. Having been inspired by his close friendship with Aaron Schwartz when the famous Internet hackativist was still alive, Lessig's charisma abounded from his entrancingly visual, rapid style of lectureship – a personal brand no one failed to mention. But outside of his panels, many knew Professor Lessig from the democracy walks that New Hampshire Rebellion organized on his behalf, in which hackathon participants had previously participated to protest institutional corruption. Ultimately, the only time they could bond with the awe-inspiring figure had been these walks, where he had nothing better to do than make small talk. Some used the

hackathon as a platform to strike a collaboration with one of his grassroots projects. That was the case of Aaron Lifshin, who got a job as the Chief Technology Officer (CTO) for Lessig's superPAC around the time of Hack for Democracy Boston. Bruce Skarin from the Campaign Con team in Chapter Six later became Lessig's right-hand on the Equal Citizens project launched during the 2016 elections, which petitioned for: citizen-funded elections; equal weigh-in in popular elections; and the end of political gerrymandering. The types of participants who attended such hackathons in the civic tech space were thus interconnected and matched their hacking activities, as we shall see in Chapter 2, with an egalitarian ethos. Their labor mirrored the virtuous conduct that Virno (1996) recognized as the basis for political action.

Of all healthcare hackathons in the Greater Boston area, the Grand Hack also welcomed the most vendors. Securing funding for such a large event was a serendipitous affair, and the MIT Hacking Group behind the annual Grand Hack at MIT had too much capital to know what to do with it. After seeing the dedicated turnout to the MIT Grand Hacks, Zen Chu founded the Hacking Medicine Institute (HMI) in 2015 to accelerate the collection of data and evidence that support the adoption of digital medicine. A senior lecturer in health care innovation at the MIT Sloan School of Management and the Harvard-MIT Health Sciences and Technology Program, Zen Chu was a serial entrepreneur and had founded Vector Ventures to fund early-stage digital health and medical device startups a few years prior. 2016 co-director Chris Lee – back then the sponsorship lead for the MIT Grand Hack I observed in April 2015, explained that the process for funding was rather informal; Zen Chu would be sitting at a board meeting of a hospital and be approached by companies who were interested in entering the healthcare innovation space. In 2015, the sponsors were General Electric (Primary Care), pharmaceutical distributor MeRk (Telehealth Track) and Microsoft (Wearable Track). Out of courtesy, the Massachusetts General Hospital's Consortium for Affordable Medical Technologies (CAMTech), which had sponsored

previous hackathons and thus had an outstanding relationship with the club, was invited to sponsor the Global Health track. Microsoft in particular had a commercial stake in the Grand Hack, having released its first-generation fitness tracker called the Microsoft Band, and needed to test its usability among physicians at the hackathon.

Be that as it may, the primary mission of the MIT Hacking Medicine was to "energize the health ecosystem to solve some of healthcare's biggest challenges by connecting the best and most diverse minds." As 2014 co-director Priya Garg emphasized in a May 5, 2015 press release, the success of the 2015 Grand Hack depended on this energy. But the mission of the Hacking Medicine Institute tells a different story: "Medical and health technologies — for decades derided as the driver of increased health costs — are now the key enabler of new products, re-imagined services and new business models to extend the reach of physicians and institutions, and achieve the triple aim of increased access, better outcomes and lower costs." Contrary to the field of journalism, for instance, the institute clearly stated that healthcare innovators had been doing the same thing for decades. There is no change in the capitalist mode of production in healthcare, only that, somehow, it had become cheaper digitally. The MIT Hacking Medicine Group might not have wanted to do things differently. As the distribution of authority inside teams in Chapter 3 showed, few teams dared to be too creative with healthcare hacks. Their activities were far less playful and followed a more rigid structure. Ms. Garg, now an associate at Boston Consulting Group, explained that diversity among technologists at these events helped innovators think more creatively about a challenge in digital health: "We were so excited to see the diversity of clinicians, engineers, developers, and designers, because we find that the best healthcare solutions come from the most diverse teams (MIT Hacking Medicine 2015)." In fact, these events were less about technological disruption than about the organizers. Participants often gravitated towards the expertise of the organizing team at these events, all of whom were

MIT mechanical engineers and PhD students in the Harvard-MIT Health Sciences and Technology program. Their record track in launching healthcare startups, talent in research, and perhaps even their youthful optimism and energy factored in some people's decision to come to the Grand Hack.

According to a community member in the organizing team, the hackathons MIT Hacking Medicine hosted could be distinguished by its direct intent to produce startups in 54 hours (Hari Iyer, interview, November 9, 2014). Despite this emphasis, sponsors were more attracted to clever applications of technology, not necessarily new businesses. With the exception of Microsoft, no one really had a stake in setting up formal social enterprises with the event participants (Currie et al. 2013; Fish et al. 2011). Athena Health MDP was an odd case. The health IT company who benefited from expanding its network of providers set up their hackathon so that teams could gain entry into its accelerator. "Plug and play" documentation, which included the instructions for using Athena Health's new API created a sandbox to limit the risks of building on top of existing code, and provided rules and mock data to safeguard client information and infrastructures from hacks. Set up this way, hackathon participants could use the API to indirectly learn about the company policies. The idea was that Athena Health could form a partnership with future formal social enterprises.

As these actor-networks show in subsequent chapters, communities of practice at hackathons did not cohere by their own structural composition, but instead were mashed up into their peculiar assemblages as participants interacted with each other, specific technologies and institutions. How they came together, moments of failure, and then recovery were specific to the individual's resistance as he or she negotiated her labor-power and intellect through language, the material constraints of the technology and lots, lots of making do. While these mashups did not always produce startups, they exhibited different degrees of consequentiality, in different

places and practices, which corresponded to different types of play labor in situations of high uncertainty that contrary to what Latour posited of assemblages, relate to certain practices in the industry in which they are held.

Resistance and Organization of the Event

In light of the arbitrary nature of the performances that these actor-networks could hold at any time and at once, skeptics of hackathons have keenly observed that the goal of accomplishing core technical work – relying on developers' already existing skills – and working synergistically, such as having another kind of challenge for novices, were not tightly coupled. Moments of resistance could be the effect of orchestrating fragmentary engagements at hackathons (McDermott and Varenne 1995). Let's keep in mind that these assemblages came to life through a mixture of virtuous conduct (Virno 1996), play labor (Terranova 2000) and the formation of the entrepreneurial subject in response to the demands of immaterial labor (Lazzarato 1996).

A hackathon is structured in such a way as to give any participant an exit. On a bioinformatics hackathon website, a "What is a hackathon?" tab describes how the ethos of the hackathon embraces resistance:

> "Hackathons are self-organizing events; any attempt to impose something as detailed as an agenda we encourage you to meet with resistance and humor. With that in mind, our schedule provides the absolute minimum amount of structure needed to ensure that something actually gets done."
>
> *Source:* GCC 2017 Conference and Hackathon Website

As the instructions stated, participants at the GCC 0217 hackathon were highly encouraged to resist any political agenda. Any attempt to fix teams, for instance, could be interpreted as against the spirit of the hackathon (Arnab, personal communications on February 25, 2017), and be met with disapproval from the participants themselves, who would complain that too rigid a structure

could hinder their creative muses. Although every organizer believed he or she held the keys to planning such events, the best structure was often no structure at all, a belief that most organizers ignored when planning the event, but sustained *during* the event.

The example of a hackathon schedule, which I pasted below, supports the principles of self-organization:

Day 0:
6pm Panel and Networking

Day 1:
9 am Lightning Talks/Breakfast
12 pm Team Assembly
2 pm Lunch
3 pm Start hacking
3 pm Mentor's workshops
6/8 pm Teams go home or move to another building to continue hacking

Day 2:
8am Breaking & Continue hacking
2 pm Submission deadline
3:30 pm Demos
5 pm Award ceremony

Figure 5 – Hackathon Schedule
Source: MIT Grand Hack Website 2015.

As indicated by the sample program of activities, the organizers of industry hackathons modeled their schedule on the many transitions their event attendees must make: between the different stages of recruiting team members, the activity of building collectively, and getting feedback on their ideas. The program was the same across hackathons, with only minor modifications by the organizers. To give participants more time on their hack, some organizers opted to have their panel of experts speak the night before, followed by one intense networking session to give participants time to network before the official opening ceremony. There was at least one pitching session at all seven hackathons regardless of the type. Although the two-hour block after pitches was dedicated to team formation, teams could form throughout the hackathon. Most

teams started hacking in the afternoon of day 1, at the same time as representatives of sponsoring companies were holding technical workshops. They continued hacking till the late morning of day 2, and demoed their prototype *en masse* at the end of the weekend.

What seemed a simple schedule was actually a formula for the induction of hackathon attendees into what the founder of the MIT Hacking Medicine Zen Chu has called, "controlled chaos, we're maximizing the number of collisions in the room" (mentor's workshop, April 24, 2015). The productivity of participants was to be measured against such friction. They were encouraged for the sake of their passion projects to form new teams if necessary. One was indeed thrown into action by having to figure out which team and project to join.

A walk-through the venue best illustrates this concept of collision, and the on-demand, affective nature of engagement at hackathons. Hacking Journalism Boston and Hacking iCorruption happened on the third floor, and Grand Hack on the fifth floor of the MIT Media Lab. Alternatively, a hackathon was hosted in the adjacent building. For example, Hack 4 Democracy Boston took place on the first floor of the MIT Stata Center instead. This last space was divided between two classrooms and an open walking area with tables and chairs, but retained its openness. Otherwise, the Open edX hackathon happened on the main floor of the company, surrounded by multiple conference rooms. Hacking Journalism NYC was hosted on the 17th floor of the World Trade Center and was similarly enclosed. Every venue however had a common area, with a projector used for lightning talks and demos. Teams would trickle into conference rooms at the corner or in the hallway as they got organized, or stay in the common area to work at one of the round tables. The glass windows that made up the walls of the conference rooms made for excellent white boards, and let curious passersby and organizers peek in. A good venue by organizers' standards had to have a transparent and circular layout to facilitate interaction and mobility, both important to boost participants' creativity and

communication. Individuals were seen to interject their thoughts on the go, in a manner that reminds us of Ingold and Vergunst's *Ways of Walking,* "in a movement that is both rhythmically resonant with movement of others around us" (2008, 2).

Figure 6 – Map of the MIT Media Lab Hackathon Space
Source: MIT Media Lab Website.

That is not to say that play labor is rooted in such phenomenological traditions (though this work is sorely needed); but it relies largely on a sense of urgency that is built throughout the event to energize the crowds. This energy was built into the hackathon mostly by enforcing time constraints. On top of this energy, hackathons followed three other rules of organization: practical education, free use, and spectacle.

I now go through them one by one:

Practical education

Participants taught each other by observing disciplinary differences at work. They also walked and talked through problems they encountered in their collaboration, drawing concepts and using their drawings as a prop to discussion. The sum of their skills, domain knowledge and

practical experiences directly played into their creativity and productivity – or, sometimes, quite the opposite – and the team composition was affected by the choice of whom to invite to a hackathon. In a few rare cases when the skill was not available among the participants, a participant would call in on their remote professional connections. The participants' networks played an important function at hackathons.

Practical education, as a corollary of play labor, thus depends on a potlatch (Mauss 2007) of skills to turn a hackathon into the pleasurable experience of digital production. Organizers' use of an application only served to bolster diversity and guaranteed an even distribution of all the skills involved software production in a given industry. The organizers of Hacking Journalism NYC used an application process asking participants to describe their design and coding skills, and specify whether they had any experience leading projects. Based on their blurbs, the organizers of Hacking Journalism assigned teams to guarantee at least one backend engineer, known to be in high demand and scarce in the journalism industry, one videographer to meet the expectations of the video theme for the event, and one project manager to lead the team. In lieu of the blurb, the MIT Hacking Medicine group required an essay application to encourage its participants to think of a creative hack for the healthcare industry and used reported statistics on its bottom survey to diversify the talent pool. The club accepted 90% of applications; of those who were rejected, most were MIT students. Having organized the event to engage external partners, a quota system was put in place to encourage students from other universities (Brown and Yale), towns (Providence, District of Columbia and San Francisco) and countries (India) to participate. This suggests that those who came from the organizer's networks were highly educated, and though loosely connected, were motivated to meet people from the same community of practice.

Team formation mirrors affinity-based networking: participants self-selected into teams and hackathons based on their interests. This is one kind of type of play labor. Given that they were complete strangers from one another, social acceptance was only secondary to their immediate needs (De Kosnick 2012, 109). Such needs included: filling gaps in technical skills, managing the cooperation and encouraging their creativity. Such needs supported existing social relations; my first impression that hackathon participants communed together out of a shared desire to hone their craft. Likewise, the technology served mainly as a device around which to recruit talent and expertise. Hackers exchanged skillsets for free, knowing they could never replace the kind of free advice and mentorship they received that weekend at the workplace. This of course required that each participant presented his or her best self if they were fortunate to have their idea draw the attention of other participants (Goffman 1956). He or she had to come up with a competitive idea that their peers would judge as wildly creative, feasible or impactful, and use his or her soft skills to motivate others to work with him or her throughout the event. Together in one room, the incoming crowd of participants could share at any given time two levels of knowledge: 1) managerial know-how from having worked in a place that specialized in the production of digital technologies; and 2) knowledge of the trends in digital labor, both of which turned any creative hack into the object of practical education.

As per the restriction on agendas, formal social enterprises (Currie et al. 2013) have minimal involvement in the activities of hackers outside of the voluntary help desk; they were there to troubleshoot any technology they made available at the event. They in turn bring back experiences at the hackathon that may be useful in their workplace.

Free to use

For technologists and hackers, the hackathon is a sacred space, untouched by corporations. The hacker ethic adopted at such events stipulated that participants must release

their source code to make it easy to share; they deemed it a good practice, should anyone outside the hackathon take it upon themselves to fix bugs or improve the hack. Likewise, technical leads inside teams took great care to remind newcomers to not concern themselves with patents. The practice of sharing code at hackathons could only *potentially* have legal and economic ramifications for the free labor that had been invested into writing it. The singular act of uploading, and copying code that already existed could boost the reputation of its author and his code, or it could do the opposite: fall into oblivion without the living labor to sustain it.

Which is not to say that programming labor remains free in perpetuity, only that the code was kept free of charge and given enough flexibility in repurposing. Code is extensible and grows in contributions when other communities of practice have maximum freedom to make it work in other contexts. In support of the hacker ethic, hackathons make it particularly difficult to transform free labor into deferred wage outside of an internship or contractual job, and seed just enough capital to build a prototype – not a company. Because they could almost never engage in production for the organizations sponsoring the event, they built the hacks for themselves. Yet, this almost never stopped them from exploring these institutional logics in appealing to their audience throughout the hackathon.

At some hackathons, participants were required to sign a video release to authorize the press to film the event in the news. At a company hackathon, more conservative sharing practices were adopted. In some cases, it was easy to infringe the proprietary rights of a piece of software the participants used to build their hack; and because disclosure of a company's know-how was exclusive to the mentee, some consent forms added a waiver clause that specified what kind of participation should be treated as private or public. Further precautions could enter into practice, such as having participants waive proprietary rights to their work at the hackathon, giving the sponsoring companies permission to review the code and test the app for the purpose

of judging, or adhere to certain guidelines, and releasing the source code in a form deemed appropriate by a certain license to be considered *"free" to use*. Under the "free to use" umbrella, all the organizers in my sample of hackathons encouraged participants to keep their work as their own, as long as they uploaded their code for review by interested parties after the event.

The argument in favor of sharing made sense on its own. The hacks, as it often was the case, embodied a projected use-value; even with a business proposal, no real exchange-value could clearly be articulated for hacks that were by nature, far removed from everyday economic realities. To be commercially viable, hacks needed to be re-designed as integral parts of a system or device and embedded in social relations that could facilitate their circulation and price-exchange in the market. The exception in my sample of hackathons was edX, the 100% open source company – legally and physically capable to implement the ideas from its contributive. Participants knew to submit their hacks for review, and to consider them on a spectrum of fully networked production.

This "free to use" legal stipulation blurred the boundaries of work, such as that participants could be motivated to build for a company, without getting bought out. If a hacker's experience at a more competitive hackathon became less positive, he would cry "exploitation!" and divert his labor towards building something of value to himself. One was free to do so because he had not been formally contracted to build the technology, but even then, one needed as much freedom as possible to iterate, pivot or change direction last-minute. The "free to use" rule was especially helpful in scenarios where technical or legal barriers of their hack would have delayed production. As such, transgression and resistance were directly embedded in the activity of hacking.

Spectacle

After skills, hackathons were a potlatch of ideas; their outcome should be no more and no less than showing what could be done. The spectacle of the hackathons was in the demos, but their formation is problematic, to say the least contradictory. Participants would structure their presentations to hide the flaws in their prototype, and to compensate, presenters would tell a compelling story for the technology they had just hacked, as to why the audience needed it to solve problems in industry. But while some of that craftiness could be captured by the demos, the unruly nature of code always mediated what could be said. Some situations could be reworked to one's advantage, and in this sense, a spectacle is a challenge one throws oneself.

Prizes, the internships, or the promise of a freelance gig added to the spectacle in lieu of an actual production of production (Wark 2004). Excess was the name of the game in the multi-sensorial deluge of information of the opening ceremony and the demos. Demos projected a reaction to their workplace conditions on-screen, and hailed technology as an equalizer. The opening ceremony drew from their co-joint emotional labor to build the energy of the moment, a sense of urgency and impact, where panel experts were then invited to empathize with their cause.

Participation was then glorified on social media. Organizers filmed a trailer and uploaded the talks and participant interviews on YouTube. Journalists – often a friend of the networks that assembled at the event, contributed to the panoptic effect of live blogging (Foucault 1995). These events helped institutions make a statement on their progress. Sponsors and host organizations interlinked online repositories and winners on their websites, claiming affiliation with their labor. This co-opting gesture is indication of the problematic nature of the spectacle. The spectacle, as Guy Debord (2015, 78) likened to science, had for function to guarantee that capitalist services would be needed by strategically staging the environment to make participants desire the

technology. But one thing about spectacles is that they could always be reprised and remade into one's own.

Mentors

The hackathon provided a means to showcase one's talent at creative problem-solving and his programming abilities amidst the unstable, rapidly changing landscape of play labor at the hackathon. Mentors, judges and sponsors participated in judging the merit of a hack, either by relating their experiences, scoring the hacks, or implementing them afterwards. In ranking hacks, judges also assessed the labor-power of a team or individual.

Organizers looked for sponsors, judges and speakers with experience launching ventures, or hacking in these industries. Their feedback raised unique challenges to embedding the hack in a community of practice. For the MIT Grand Hack, sponsors placed mentors and wrote the judging criteria. At other hackathons, they could help organizers select the judges by each recommending three people as long as they were separate from the company. Judges selected by the sponsor's network and a proxy for potential investors, were only allowed to evaluate and not buy the labor at the hackathon. With their commentary, they communicated the kind of work they valued in industry.

The Grand Hack and Hacking Journalism positioned mentors to be accessible, either at the entrance of one of the large meeting rooms, or by having all workshops take place at a non-disruptive hour (3:30 to 5pm on day 1). The MIT Hacking Medicine student club discussed their conditions for rejecting mentors at their weekly meeting on March 30th, 2015. A mentor had to give honest feedback, and because it was hard to filter for agendas, the organizers decided that their founder would bring five people he trusted, and that everyone would recommend the rest. Mostly they were chosen based on their reputation as experts in a related domain.

There were different models for mentorship. At Hacking iCorruption, the Harvard Center

for Ethics Research Fellow were the go-to lead on the project; on such an occasion, the fellow served as the mentor. The teams at Open edX had at least one edX employee to point them to existing resources at the company. Meanwhile, the journalism hackathons devoted the early afternoon hours on Day 1 to teaching participants how to code some examples for the devices they had at the event. Some example workshops are given here for Hacking Journalism NYC:

> **Andrew Montalenti from Parse.ly:** Parsely helps publishers understand how readers are responding to content. Showed which analytics actually matter, and how you can leverage it for engaging content.
>
> **Andy Pellett from Embed.ly:** Demonstrated how to easily include images, videos, and rich media in your apps. Intro tutorials and demos on working with the Embedly API, Cards, and Player.js for rendering media and controlling video in your apps.
>
> **Scott Money from Ramp:** Offered a showcase of interactive video experiences and showed how to implement them with RAMP's APIs. Ramp builds products that use video to improve internal communication and marketing initiatives.
>
> **Rashmi Raman from Bloomberg:** A quick turnaround to build prototypes can improve the creative process. Showed demos and tutorials for working with Brisket.
> Michael Sagalyn & Morgan Timpson from IBM: Gave tutorials and demos on accessing Watson Analytics. Watson helps sift, analyze, and visualize datasets.

Figure 7 – Technical Mentors' Introduction and Workshop Themes
Source: Matt Carroll's Blog 2014.

The mentors were most useful three hours before teams had to demo, between the hours of 11 am on Day 2 and 3 pm. At this late stage of hacking, the kind of feedback mentors gave was most pertinent to the hypothetical product life cycle of a technology and helped the team anticipate questions from judges.

Most projects had to be submitted by noon or 2 pm on the last day when the teams were frantically patching their pieces of code and practicing their presentations. Demos took place between 3:30 and 5 pm. In five to ten minutes, the teams told the story of how an imaginary, industrious worker wanted to get her hands on their technology to solve a problem in her line of

work. Demos demonstrated how the hack worked, and finished with an extensive list of the experts and supporters who had made it possible. The decision of whether to give prizes based on merit depended on the type of hackathon. Hacking Journalism NYC, Open edX, Hack4Democracy had neither judges nor prizes. The crowd of journalists tweeted their favorite hacks, and the civic hacks at Hack 4 Democracy were upvoted by participating activists. The audience at edX just listened. "To have prizes, would near insult," explained the first edX hackathon organizer (Ned Batchelder, interview, November 22, 2014).

Conversely, Hacking iCorruption, Hacking Journalism Boston and Athena Health MDP offered prizes, corresponding to increased trickster activity. It was not uncommon for experienced hackathon participants to game that prize system. Two teams at journalism and healthcare hackathons hacked a particular device because no one else was, taking advantage of these odds to guarantee themselves the hardware prizes. To counter this movement, I once saw a mentor with a double agenda try and increase the odds that participants use his API in order to raise a fairer assessment. The enforcement of *alea* (the element of chance in the game) by the organizers, mentors and judges conflicted with the strategic compulsion to monopolize a technology effective in *agon* (the competitive element of games); the productive friction between these ideal forms of play propelled teams to be more creative in relation to their peers (Caillois 1958). Such unintended consequences of self-organization did not always please the organizers, but they made them proud; it reminded them that they could also be "punked" out of their planning for more communal experiences. While engaging in this struggle for capital, participants could also decide to ignore their bad odds if they were excited about a new feature; many teams, for instance, tried to hack the Microsoft Band and the GE Portable Ultrasound at the MIT Grand Hack because they felt they should prepare in the advent this new technology gained traction in the future.

Only three teams could be recognized at a single event, or for a single track. Such formal recognition only complemented the more pleasurable aspect of competition with the other dozen teams. Contributive and communal hackathons support my thesis that hackathon participation was a form of mock-networked production without strong evidence of its impact on waged labor. Hacker teams at catalytic hackathons might have been eligible to win the equivalent of one-month salary ($1,000- 3000), but it was a small token of recognition in comparison to a larger package including office space and phone calls with a company).

Though sponsors were prohibited during all of seven hackathons from influencing the judges, the MIT Grand Hack let Word sponsors (donated $3,000 to the event) and Co-Hosts (donated $5,000) pitch and interview participants. Sponsors worked together with hackathon organizers to write culturally-relevant criteria. GE explicitly valued hacks that could integrate ultrasound technology in the existing clinical workflow. MeErk, a pharmaceutical company that has since merged with numerous US companies and changed its name, was looking for telehealth interventions. All hacks were judged on the criteria of innovation and business viability.

> What it is? $1000 for 1st Place (1x) | $500 for Runner-Ups (2x) for each of the 4 tracks
> Who will decide? The track prizes for each track will be decided by 4-5 diverse judges with expertise in that particular area.
> How they will decide? The track prizes will be awarded based on how well the team addressed the following 4 items:
> 1. Health Impact – How large of an impact will this solution have if it is implemented? Does it solve an important health problem?
> 2. Technological Innovation – How technologically innovative and creative is the solution? Does it have a "creative application of ingenuity"?
> 3. Sustainable Business Model – Does the project have a sustainable business model?
> 4. Presentation – How effective was the presentation? Does this team have what it takes to carry on the project and implement it?

Figure 8 – Grand Hack Criteria for Hudging Hacks in All Tracks

Source: Grand Hack Website 2015.

Judges only included one representative from each sponsoring company. The process of

establishing the value of a hack for them had to be kept impartial.

Who won was more telling of how subjective these criteria were. Judges preferred to see real implementations using existing technology. The winning hacks in the Telehealth and Wearables tracks had focused on integrating two devices made available by the sponsors at the event: the GE Portable Ultrasound and the Microsoft Band. Most of the hacks gained recognition for designing a creative use case for these devices, and demonstrating how to embed the technology within the healthcare ecosystem of a specific disease or type of patient.

Given that civic hacks were based on previous research conducted on forms of institutional corruption, the value of each hack was partially framed in terms of exposing a "flaw in the system." Participant judgment and judge commentary both reflected a preference for this narrative arc. The subject of incentives in particular, and citizen education, were often brought up in the scoring rubrics.

The comments for the LIBOR Alt Repair, CampaignCon and WeCott hacks were copied and pasted in Appendix A, along with the five judges' scores under five criteria, to show how the civic technologists and activists' communities of practice made sense of their value. I now give examples.

The judges that graded LIBOR Alt Repair among other civic hacks included two journalists, one city council member, one professor of business, and one professor of government. They did not represent the e-government industry, but formed such a well-rounded panel that one participant commented on how clever this arrangement was, by stating that "journalists in a way are the best people to judge an anti-corruption hack."

The criteria for assessment fell under the five categories of APPLICABILITY, NECESSITY/IMPACT, PRACTICALITY, OPENNESS or REPLICATION, and ACCESSIBILITY. Most hacks scored similarly except for BillFinity, SchoolHouse Ethics and

LIBOR Alt Repair. Their median scores were most variable under IMPACT and OPENNESS (Appendix A). When I asked before and after the demos, the judges confirmed that the criteria were clear to them. From the pattern in scoring and commentary, I comment on the judge's stylistics. Judge 1 did not score but offered critiques. Judge 2 took copious notes. Judge 3 was by far the harshest grader, with an average on 3.08 (out of 5) on five measures for all ten hacks, and was especially critical of the capacity for these hacks to fulfill the NECESSITY/IMPACT (incidentally, framed as the question "Is the problem well-defined?") and ACCESSIBILITY criteria ("Is the solution user-friendly?"). Judge 4 only scored but did not comment. Judge 5 either loved or hated the hack, grading either generously or harshly. Judge 6 focused on the PRACTICALITY criterion. The different emphases made for a diversity of opinions that nevertheless pointed to the near impossibility of reaching consensus on the definitions of institutional corruption, which of its problematic cultural practices had priority, and which characteristic of the hack should make the mark.

The LIBOR Alt Repair scored highest under APPLICABILITY and PRACTICALITY, but low on IMPACT and ACCESSIBILITY, on the grounds that users would have to request that this rate be used, and that even if they did use it, lenders would not be motivated to reciprocate. Judge 2 especially liked the idea; this anonymous person thought the website explained the concept very well under ACCESSIBILITY, but under OPENNESS/REPLICATION, contradicted herself, writing that she was "not sure" next to the question "How forkable is the source code?" in the form. Judge 6, although we know by now focused on answering whether the hacks were solving a problem in institutional corruption (APPLICABILITY), had an adverse reaction to LIBOR Alt + Repair and wrote a critical comment under NECESSITY/IMPACT "Theory of change very unclear," on top of disagreeing with Judge 2 under ACCESSIBILITY, "Not sure who would use this…" Based on the two judgments by Judges 2 and 6, I conclude that

there was no consensus on how *accessible* the technology was. Lilia Kilburn (2015) blogged at the event about the *practicality* of such hack, quoting Silz-Carson's statement that banks' strategic manipulation of LIBOR rates likely affected "many of you in this room," through student loans and mortgages. The summary of the judges' comments indicate that the hack was most applicable, but that much bigger concerns were left unanswered by such a creative hack concerning its competition. Since LIBOR was so widespread in society, who would be incentivized to use an alternative rate?

Like LIBOR Alt + Repair, Campaign Con was rated variably. Judge 5 "loved it" and gave it all 5's, which explains how the hack placed third. Judge 7, however, did not understand that the data used was public and scored it low on OPENNESS and ACCESSIBILITY. The critical judge also gave it a 3, somewhat low, on PRACTICALITY, only to have Judge 2 disagree and give that measure a 5 out 5. In addition to ACCESSIBILITY, Campaign Con was given an impartial 3 by Judge 2 on APPLICABILITY, reflecting Judge 1's confusion about the kind of technology and issue on which the hack was built, and could be attributed to the combination of the team's diverse interests in both an audit tool and data visualizations. Together, these comments implied that the judges looked for a simplistic design that accounted for the user's learning curve upon using it for the first time.

WeCott was an example of a platform for social movements, and can serve as a proxy for the evaluation that Townhall would have gotten if there had been judges at Hack 4 Democracy. WeCott scored high on all measures. Judges felt that the platform, fully built to publicly pledge the amount users will boycott on social media each week, ultimately reducing the transaction costs of participating in boycotts, was "very easy to use," with "an elegant design" and "well-thought out." Only two words of caution were expressed. Judges 4 scored exceptionally, giving the hack only two points on necessity and impact; Judge 5 gave WeCott a solid 4 on

PRACTICALITY but voiced his concern for "people coming up with boycotts that should not be started." The feedback for WeCott suggests that the judges viewed fully functional hacks favorably, and preferred it to be built by the end of the hackathon to showcase its feasibility. In terms of platforms, the panel of academics responded positively to a well-thought-out design and it also suggest that they had been taught to care for an intuitive design that addressed the user experience.

Contrary to the wisdom of the crowd, who felt that as long as the proof of concept was easy to grasp the elegance of a hack was only a bonus, the judges' comments seemed to suggest otherwise. While the objects of judgment were the concepts themselves, the game of logic the judges played inadvertently reflected back on the team's technical prowess. To be able to build or submit oneself to the rigors of creative problem-solving, the teams in these six chapters had to draw from complementary domain knowledge (The Cuff Guys), expertise in a programming language (Sirona Care), or experience solving the same problem (Townhall). Coincidentally, these were teams were fairly homogeneous with a tangible measure of expertise in programming. This emphasis was very apropos when one took into account the social life of these hacks (see last section).

The winning teams in my sample were: Sirona Care, Ultrasound Training (both 2015 Grand Hack winners), and The Cuff Guys (a 2014 Athena Health MDP runner-up). These results could be attributed to the fact that only healthcare hackathons were competitive and awarded prizes. Symbolic prizes and the evaluation criteria for awarding them were but one way to valorize hacks. The civic hackathons did not give prizes and used different methods to communicate the merit of each hack. Since the civic and journalism events attracted strong communities rather than loose affinity networks, implementation was the best measure of its worth. Post-event leads were also a measure of success for the participants at Hacking

iCorruption; for instance, Silz-Carson was contacted by a professor at the Los Alamos National Lab about transforming the LIBOR Alt + Repair hack into a summer project. Similarly for the journalism hackathon in NYC, I know of at least one videographer who got a lead for a contractual position at the hackathon.

How much does it cost?

Representing institutions at a hackathon ironically had to pay to not play, but they also assigned merit according to the needs they saw in their sector.

The university hackathon student organizer network, a nonprofit by the name of Major Hacking League, estimated costs for a hackathon range from $2,000 to $10,000 (Table 7). The organizers of the first journalism hackathon and Athena Health confirmed these estimates. Institutions hosting this type of event had no trouble securing funds. The Center for Ethics at Harvard Law School, for example, self-funded its own event and paid for their space at MIT Media Lab. As much as $60,000 was spent hosting the largest hackathon in my sample (400+ participants).

Table 7 – Breakdown of Expenses for a Hackathon

Venue: $0-3,000

This cost can be factored out when the hackathon takes place on the organizers' campus; in most cases, Harvard and MIT need student affiliates to book a room at each other's campuses and pay a fee to rent the Barthos theater for their symposia: $100/half-hour

AV Equipment: $900 to stream in four rooms

Food (2 breakfasts, 2 lunches, 1 dinner and drinks): $1,500-6,000 ($15 per attendee)

Prizes: $0-26,000

MIT Grand Hack: 26 prizes x ($500-$1,000) = $13,000-26,000

Devices given free of charge by the sponsors if willing

Community-oriented hackathons (Hacking Journalism; Hacking iCorruption and Hack4Democracy did not have prizes = $0

Swag (small gifts with club or company logo): $0-500

MIT made T-shirts with its own logo although student clubs cannot use the MIT name.

Travel expenses for speakers: $1,000 (occasionally, most speakers are from the area)

Sources: Hacking Medicine 2015; Major League Hacking 2015.

Table 7 is a breakdown of expenses for the MIT Grand Hack. For most self-funded events, food and venue were the only big expenses. For more expensive hackathons, organizers had to restrict, but also offer, more opportunities for gold sponsors to get involved with participants. Gold sponsors each put in $15,000 for the Grand Hack and offered prizes, but as Major League Hacking stated at its 2015 annual organizer conference, this large sum was little compared to the costs of hiring a single employee at a large firm, with much more risk associated with a wrong match if she were to quit.

The instinct to separate hackathons from any financial incentive, or path to commoditization, is further indication that participation should not feel *like work*. This gesture was a difficult one to fully endorse, as sponsorship could also be a source of deeper engagement. In a pay-to-play mode, levels of sponsorship determined what sponsors might bring to the event, and perks, such as the ability to hire a participant after the event (Table 8).

Table 8 below shows how sponsors were allocated different obligations and benefits by the monetary range in which they opted to finance a hackathon. The companies who contributed a Byte ($2,000) – mostly software companies – were allowed early access to the resume book. All the sponsors were asked to contribute an API[11] for use at the hackathon and to send mentors to train interested developers. In exchange, the sponsors were allowed a company table and marketing materials at the event. In some cases, more generous sponsors (Co-hosts for $5,000 and Words for $3000) were placed as judges to evaluate the merit of each hack that used their technology. It should be clarified that resources companies placed at the hackathon were

[11] Application program interface (API) is a set of routines, protocols, and tools for building software applications. An API specifies how software components should interact.

proprietary and only made available at the discretion of the team and its mentor.

Such a matrix of exchange safeguarded participants from the possibility that sponsors might want to copyright and, in turn, *mis*-credit the author of a piece of code. Participants were held to the same restrictions, and were invited to assess the impact that their labor in relation to *ideas* in lieu of a deferred wage (Beller 2016). In order for the craziest ideas to come to life, their creativity could not be corrupted by the desire for a more commercially viable hack, despite its seduction.

Table 8 – Sponsorship Levels

	Bit $500	Nibble $1,000	Byte $2,000	Word $3,000	Co-host $5,000
General					
Company table	X	X	X	X	X
Send mentors	X	X	X	X	X
Bring an api	X	X	X	X	X
Bring hardware		X	X	X	X
Host an approved event/tech talk			X	X	X
Sponsor a prize			X	X	X
Place on judging panel				X	X
Recruitment					
Collect resumes during the hackathon	X	X	X	X	X
Access to attendee info after the event	X	X	X	X	X
Access to resume book after the event		X	X	X	X
Early access to attendee info and resume book			X	X	X
Recruitment pitch at opening ceremony				X	X
Room to hold interviews during hackathon				X	X
Branding					
Company logo on t-shirts and marketing material	X	X	X	X	X
Company swag in welcome bags		X	X	X	X
Be a "materials" sponsor (filament, wood, acrylic)			X	X	X
Be a "meals" sponsor				X	X
Event co-host					X
10-minute keynote					X

Source: Major Hacking League 2015.

Life after the hackathon

Like Conrad of Ultrasound Training reminded his team and the GE judge sitting during happy hour, the judges cared about the human capital – whether they had the right distribution of expertise to implement the hack after the event. Any update to a hack was purely incidental for the most part, and we see in the cases of Pill Pack from the 2014 Grand Hack, CampaignCon, The Cuff Guys, Video Pizza and LIBOR Alt + Repair that implementation was the exception.

The first obstacle was incompatibility with the current climate for innovation. In her follow-up survey, Professor Silz-Carson responded to the feedback she received during the conference on Institutional Corruption two months later, where she manned a table with James to answer questions about her hack. Regarding the lack of incentive to adopt, she replied that she did not think people would use alternative benchmark rates until another major bank crisis of the scale in 2008 happened again.

Pill Pack was the gem of Hacking Medicine, and the only startup to come out of their startup weekend. In this situation, the legality of the hack was called into question. On April 14, 2016, *The Boston Globe* published an article regarding Pill Pack's recent dispute regarding its drug plan. Pill Pack was the mail-order pharmacy hack, originally conceived at Spring 2014 Grand Hack, which had raised $62.5 million in a series A. Operating out of a pharmacy in Manchester, New Hampshire, PillPack sends out medications in presorted packets that prominently show the time and day they should be taken, a system designed to help people better manage multiple prescriptions. The paperwork dispute was an allegation that PillPack had filled out Express Scripts partner's paperwork incorrectly, and did not follow consumer safety regulations. Highly regulated sectors like pharmacy, which require multiple licenses and approvals before they can distribute; rumors that PillPack was not licensed to sell in certain states and had illegally delivered some prescriptions met with some rebuke from the startup, who

suspected Express Scripts of wanting to maintain its monopoly on mail-order pharmacies. Two weeks later, *The Boston Globe* announced PillPack had resolved the dispute and was working on a final contract with its major partner.

CampaignCon was not a revolutionary hack, according to peers at Hacking iCorruption. But its charismatic leader Dhrumil eventually finished coding the project on his own and collaborated with a representative of the FEC on an added feature. Part of the reason why the rest of the team did not continue with him was that its members had not found significant revisions to the amount of campaign donations to publicly denounce any shenanigan. Dhrumil, who had grown addicted to hackathons, attended Hack for Congress in DC, where he pitched CampaignCon to a member of the FEC staff. The FEC had apparently been looking to audit their upcoming API and gladly worked with him to finish coding the hack. Five months later, Dhrumil met with me at a coffee shop in New York and expressed a continuing interest in CampaignCon. However, the project was not a priority and he continued to procrastinate on it. This case of CampaignCon is instructive in that regard: what needed to happen after a hack was built into a fully functional prototype to launch it as a startup? Dhrumil had perhaps been the only one invested in CampaignCon. As Bruce (interview, October 2, 2015) and Max (interview, June 24, 2015) later explained, Dhrumil had complete ownership of the hack, an important factor in continuing. To put it simply, Dhrumil did not express any interest in launching it as a startup (interview, May 1, 2015; interview, September 26, 2015) – a sign that he neither cared, nor found enough public interest to want to quit his job at political news agency to scale the hack.

The pair of developers behind Video Pizza met regularly, and also finished coding their hack. According to Ross Goodwin, he and Daniel McLaughlin from *The Boston Globe* were best friends and driven by the desire to find a new approach to coding the video slicing technology (interview, June 11, 2015). The social lives of CampaignCon and Video Pizza confirm Conrad's

(pers. comm., April 25, 2015) suspicion that the reason why judges awarded the money prizes to the team with all the right bundles of skills was because they would not have to go through the trouble of hiring more talent outside the hackathon. Keeping a team together was hard work, and no one really wanted to work for free and pour their blood money into transforming the hack into a company that risked failing anyway.

Prizes actually de-incentivized Sirona Care. The runner-up team received $500 and a fixed number of mentor hours from MeRck. Aditi and Ankita had e-mailed all the other members of their team right after the Grand Hack inquiring about their commitments, and received notice that Priya, Katrine, Kamilla and Joyce, all sophomores at MIT, could only commit to a monthly short meeting. Ankita decided then that they needed more volunteer hours than anyone on the team was willing, or capable to give at the present time. Before getting back to their own schooling, Aditi and Ankita made sure to take advantage of the mentor hours offered. MeErk, a telehealth company originally based in Switzerland, had offered the most attractive prize to Grand Hack participants, and many had listed on their application to the hackathon that they had hoped to score an internship with them.

A similar story can be related from The Cuff Guys, who suffered for logistical reasons,. Since they had been awarded a grander prize, they committed to at least try building their hack after delivering a low-tech, winning Powerpoint presentation at the hackathon. Athena Health had taken six months to disburse their $1,000 cash prize, which Andrew had calculated was barely enough to build a prototype. Despite these setbacks, the team of three traveled regularly from different cities to take advantage of the six-month space in Athena Health's accelerator program. They scheduled their meetings whenever one was in town to pitch The Cuff Guys at other health hackathons. Judging from its reception as a relatively older hack, Andrew tried to join other projects for a different experience. Khalil also showed signs of its commitment waning

when he joined the MIT Hacking Medicine Club to organize the Grand Hack, which kept him too busy to participate at any more hackathons.

The symbolic prizes were therefore a good indicator of whether the team would continue working on the hack for a few more hours, but they were hardly motivating in the long run. Most students in the teams did not want to commit to scaling their hack. One major barrier to commercializing a hack (if that was the team's end goal) would have been the relative lack of business-savvy and missing technical expertise among the engineers and developers to transform the hack into a product and company. Most teams that continued (The Cuff Guys, Reciprocal, CampaignCon, and Video Pizza) had only one or two members most invested in the idea, and lost most of their team members after the hackathon.

Ultimately, the beneficiary of a hackathon might be the organizer. In the case of Hacking Journalism and Hacking iCorruption, the main organizer (a nonprofit or small company) increased its visibility by teaming up with MIT. In the case of the Grand Hack and the Open edX, sponsors likewise improved MIT's branding.

Chapter 1 Conclusion

A number of hack do's and don'ts have emerged as hackathons became more sophisticated. Of the few rules enforced at the 2015 MIT Grand Hack (Figures 9 and 10), five hack do's and four hack do not's encouraged the audience to concentrate on a pain point; not get stuck in one cycle of iteration (or with one team); and in this regard, take advantage of the "controlled chaos" at the event, a phrasing repeatedly used during the opening remarks.

Controlled chaos is a synonym for "multitude." The demographics of the hackathon indicate that the hackathons formed a diverse intelligentsia. Its physical form was the basis for political action.

These hack do's and do not's can be conceived as giving participants a common language to transgress their corporate mold. Once internalized, these rules guided negotiations that took place among participants during the team formation stage. Some first-timers were surprised; they could float around the room – *that* would have been considered sensible! Conrad walked out of a team with two others, and the Video Pizza team agreed to merge with a group of participants when they realized two teams had the same idea, but one the better atmosphere. In fostering and breaking ties in cooperation, this dissertation will show that these assemblages cohere because they are productive. This virtuoso has to be built his authority from the ground up.

Such resistance to being told what to do is part of the Hack Do's and Hack Do not's, which amplified the impermanence of social ties and disregarded any vertical leadership. "Openness" was the name of the game.

It is interesting to see the first two hack do's: "Open yourselves to crazy ideas" and "Seek diverse teams." Because most teams formed around the interests of their members, and some otherwise could not balance their skillsets. The first axiom was untenable in some cases. Being able to do so was a matter of learning how to creatively solve for one's lack of skills, and knowing how to engage others' creative faculties. This free-form style of collaboration *trusted* such volunteers to know where his practical interests and skills fit and gave them more autonomy. Participants who were not interested in the same set of questions, did not have their emotional needs met, or disagreed with their team's approach, were encouraged to hop from team to team. This unstable hierarchy was designed to orchestrate conflict, so that participants could "fail fast and pivot." (Priya Garg, pers. comm. April 24, 2015). In this game, diversity and creativity were assets in one's practical education.

Figure 9 – Hack Do's

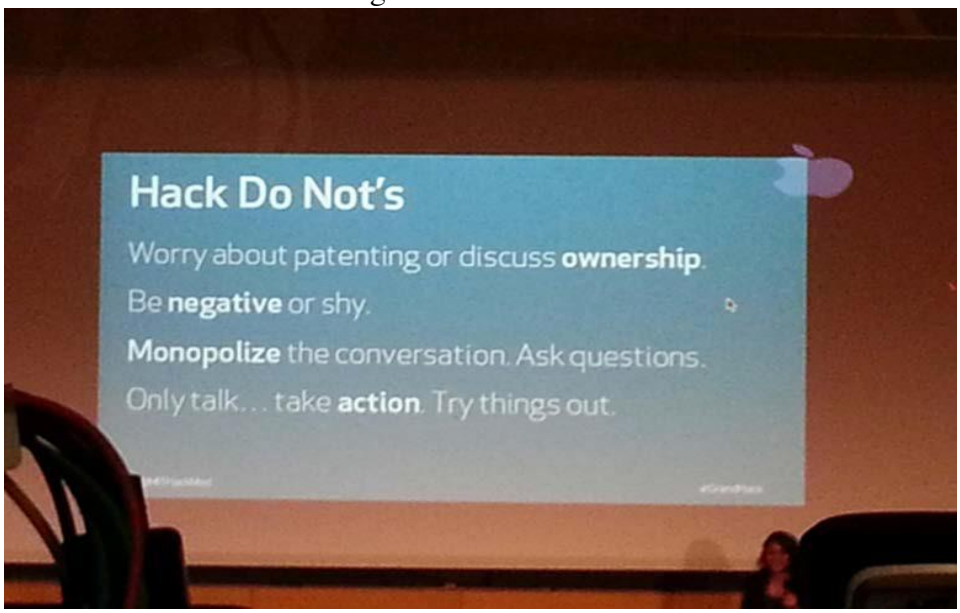

Figure 10 – Hack Don'ts
Source: MIT Grand Hack Opening Ceremony 2015.

By consenting to make the product of their weekend collaboration "free to use," an alternative legal framework was set in place to remove participation from the constraints of copyright that allowed creative modifications to software libraries. Hackathon organizers restricted sponsors from buying any code. A programmer could, but rarely, be hired after the hackathon by a sponsor. Aside from the prize system – a property of competitive hackathons, the

relative absence of financial incentives opened up the floor for play. Those looking to win a prize gamed the system out of principle: to prove that they could. The identification of a flaw in a system's design – including the organizers' – made for some interesting moments in which participants readily engaged in political action for fun. The virtuoso knows his craft just enough to transcend it, but it needs the structure of the spectacle and an audience to do this.

The focus on building under a short timeframe and the advertised deluge of ideas contributed to the energy and the spectacle of the hackathon. The spectacle immersed participants in the possibilities advanced by novel approaches and methods to working with technology (Dean 2012). Certain hacks could be reshared and improved post-hackathon, but in the small likelihood that it did make a statement at the event, the judges' commentary indicates that this glorified form of cultural production is much more about valorizing a team's labor-power rather than the hacks themselves. This assessment was extended through the simulating the fast-paced, iterative cycle of software development, for which various levels of adoption allowed for timely, personalized feedback on prototypes. Having no such restraint as those that concern modern-day corporations, the multitude was free to play, hack their own mode of production, and respond to its demand by fully asserting its subjectivities.

This virtuous conduct would for this very reason partially challenge Debord's thesis of a society on spectacle as the fulfillment of desire and the capture of value. With one caveat: the hackathon may to some degree illustrate pre-capitalist modes of production but in Section II, individuals have shown pre-made allegiances to the values and practices of their professional communities. They have made themselves singular through their play, but as Virno posited, some of their political actions are not completely unrepeatable.

Section II: Methodological notes for the Study of Play in the Struggle for Work

Section II, which includes Chapters 2, 3 and 4, delves into the transcripts of collaborations that took place at hackathons in the sectors of government (Chapter 2), healthcare (Chapter 3) and journalism (Chapter 4). For each industry, two teams – one for each of two hackathon in one industry in my sample – illustrated different types of play labor in relationship to the technologies they hacked.

My goals for this section are two-fold. As I was trying to figure out a transcript style, it made sense to have a team's pivotal moment be the focus of Chapters 2 through 4. I offer the case studies of two teams to account for differences in the organization of play labor that are hackathon-specific, and end up with my own typology of different play labors. Play can be studied in relation to specific work practices that each team brings from the industry in which these hackathons were held. I became increasingly aware through these transcripts that half of the conflicts arose from technical constraints that challenged a participant or team's assumptions. The deliberations that ensued were informative in highlighting which practice needed to change.

Secondly, I periodically weave linguistic anthropological notes to frame language use within the context of these brainstorms, and relate it to the code lying in the team's collaborative repository. I start to examine processes by which utterances are produced to cohere with the resultant code. This analysis is important for distinguishing between software production and language use. I reference Lucey and McEney's (2017, 10) model for situating language use, in seeking "indexical relations between instances of language use and systems of value and meaning that are sociocultural in nature and immanent in the social context of the instance of language use." Here, systems of value can encapsulate the very demands of technology. That is, the context of software production emerges out of interaction or better yet, when it is asserted by

the speaker out of a myriad of possible contextual constructs that are negotiated as the coming work practice in that sector. That these technologies were adopted at work two years later proves that the teams' imagination worked effectively to reflect market demand for next year's trends.

It should not come as a surprise that some utterances are entangled in code and vice versa. Humans and machines require their own separate syntax to read utterances; machines sometimes require an additional step. The instantiation of code are nevertheless the result of two forces of production, social and material. In this dual model of production, participants are shown to apply their interests and judgment to normalize (Debord 2015) their actions, which must necessarily come from their work experiences as programmers, journalists and doctors. In this vein, the hackathon offered them a space to articulate some of the daily frustrations in their workflow, and excitement for new developments in their fields, i.e. machine learning techniques and video in journalism; social media in e-government; and wearables and telehealth in digital medicine.

Each example is a specific instance of play labor, for a total of six types in the next three chapters. Chapter 2, for instance, drew from activists' humor and a simulation game (moved to Chapter 5) to manage cooperation and creatively solve a problem in institutional corruption. Chapter 3 gives accounts of how siloed work was playfully accounted for by a rigid division of labor, how a team "pivoted" on a hack to better match their expertise, and how a role reversal defied the assumptions of a third team. Chapter 4 deals with language-specific differences in programming, shifting the authorities of certain members of journalistic teams. These examples show that cooperation is mainly managed by play, but as I argue below, this play is also a part of a larger cultural framework for work.

The first indication that to me, play is work, happened when I was cross-referencing all

the pitches and discovering that teams preferred a certain technology over another depending on the industry that organized the hackathon. Teams at journalism hackathons relied heavily on machine learning techniques to write and produce the news, while the healthcare teams worked on hardware problems. Civic hackers were looking to contribute audit tools or build their own platforms to improve their communication. These patterns become more obvious when I share the list of all hacks demoed at each hackathon in the following chapters. Additionally, I supplemented these findings with archival research on industry blogs and professional listservs, which kept me abreast of trends and issues – and work practices – in the industries represented at hackathons. Together with the pitches, these trends are reflected in the examples.

The study of misunderstanding and diversion color the play illustrated in those vignettes, but there are obviously limitations. When participants talked about reshaping media use for their own needs and purposes, they were also coding, though not always at the same time. I only have access to the code uploaded on an online repository platform called Github, which I know does not store all the code and only a few updates. The code repository lacks some of the corrections that we would see if I had been able to film both the speaker and the screen as she coded. Silences in the transcripts offer some insights in the function of talk inside teams. Those intent on working did so silently after a brainstorm, only speaking when a problem arose that required the mediation of other experts. Others yet talked only to manage their cooperation. Often, these forms of making do were an invitation to play. Talking and making resisted cultural flows of digital labor by bringing awareness to its excesses. Participants did not condone them but worked with, or around them as they found ways to creatively solve for issues in their workflow.

Finally, the examples in this section will sometimes make play labor a kind of intellectual labor. Negri and Hardt's (2000, 108) identified two types of immaterial labor: "affective labor,"

which dominated forms of belonging at the hackathon, and intellectual (primarily linguistic) labor, which produces ideas, symbols, codes, texts, linguistic figures, and images. I show how the two are intertwined in the first case study in Chapter Three. Conjoint intellectual labor was actively managed via one type of play labor – humor.

Chapter 2 – Managing cooperation in e-government

In Drouhardt et al.'s (2017) typology, civic hackathons are of the contributive, almost communal type. Most governments rely on external partners to develop the technology to provide information to voters. Which is not to say that most contributive hackathons have a particular cause that they advocate to hackers, only that most civic hackathons, including Hack 4 Democracy (H4D) and Hacking iCorruption in this chapter, regularly mobilized hackers who already supported a cause, be it transparency in the elections or environmental justice. This chapter is exclusively devoted to campaign finance reform.

I outline the first kind of play labor to emerge from two scenes in which one group of activists called Townhall deliberated on the nature of the sociotechnical issues they faced. The developers and team leader Ralph reached an impasse that the latter effectively mitigated through humor. Unlike Drouhard et al.'s (2017) typology, no defined task had propelled the Townhall hack, but it did share some commonalities with the contributive type of hackathon: a desire to make an impact had motivated Devers and Jeremy to join best friends Dan Porter and Ralph to build this hack, and a shared concern for the relative absence of multi-scale decision tools in e-government had qualified their mode of participation. This team of four had not come to learn like some members of the Campaign Con team in Chapter 6 did, but instead their participation solely exhibits the kinds of virtuous conduct – as end in itself, members of a longstanding community of practice are only acting out of a belief that they are doing the right thing – for which multiple sensibilities produced the value-form of the Townhall hack.

The hackathon that this team attended was the famous Hack 4 Democracy series, whose third event took place in Cambridge, Massachusetts. H4D3 took advantage of the long history of reform in the New England area. I now share its background.

Structure and historical backdrop of civic hackathons in Boston

Hack 4 Democracy was preceded by the Citizens Rising Anti-Corruption symposium, which opened with Harvard Law Professor Lawrence Lessig and Princeton Professor of Economics Martin Gilens presenting their research. Lessig defined Tweedism the corrupting influence of one dominant group of donors in US politics. Electoral campaigns, he argued, are set up to be self-defeating, with thirty to seventy percent of candidates' time spent fundraising.

> Dialing for dollars, calling people they've never met before to ask them for the money they need for their campaign. As they do this, they *learn* something. They learn which buttons they must push to get the sustenance they need, to make it so their campaign survives. B.F. Skinner gave us this image of the Skinner Box, where every stupid animal could learn the buttons it needs to push in order to get the sustenance it needs. This is a picture of the modern American *Congress* person, [audience laughs] as he or she learns which buttons must be pushed.
>
> *Source:* Lawrence Lessig (speech, Boston, September 19, 2014)

The goal of the first civic hackathon in Boston was to bring attention to the issue of money in politics. A year earlier, Aaron Lifshin and Sarah Bonk had co-founded Team Democracy to connect technologists with short-term opportunities at civic organizations. They had met at a conference and been surprised at how difficult it had been for technologists like themselves to contribute to a cause. The actor-network at Hack 4 Democracy included several political nonprofits advocating for campaign finance reform: The New Hampshire Rebellion, founded by Professor Lessig to raise awareness about the issue by organizing marches twice yearly; Represent.us headquartered in Florence, MA, for which two chapters at Harvard and MIT brought many student organizers together at this event; MAYDAY.us, Professor Lessig's superPAC fundraising for a constitutional amendment; Democracy Matters, which trained student activists at leading US college students; and Open Democracy, an independent news website focused on politics. According to one organizer whom I interviewed two weeks later, the

visions of these nonprofits surprisingly failed to align at the hackathon, indicating that structural factors (i.e. projects under way) might have been difficult to import at the hackathon in a free-for-all format (Daniel Wong, interview, November 10, 2014).

Contributive hackathons seek the continuity of projects, and scalability[12] was of the highest priority at H4D. The five organizations pitched infrastructural needs. MayDay Labs looked to hire a freelance developer. A deferred wage in this case required someone to commit to working on a project for several months, without benefits and no opportunity to join the PAC on a more permanent basis.

Civic hackathons differed from the journalism and healthcare hackathons in their insistence that contributions be made directly to participating organizations, not an imagined user. This insight contradicted my original hypothesis – that participants played to build their hacks and often did not receive contracts or advances at the event. It compelled me as an ethnographer to follow the organizations and their freelancers for two months after H4D3, to see where the money went.

Prior to the event and again after the symposium, the following pitches were made by the nonprofits on the hackpad and repeated after the symposium:

- "Need infrastructure to facilitate this campaign"
- "How to identify influencers to spread message"
- "Build a tool that enables us to mobilize 50,000 people in NH."

Based on the language of these pitches, it was not clear how a certain infrastructure could help manage the logistics of the walk. One nonprofit suggested a supplementary mobile app. Another pitched an app that would allow people to find shelter by themselves during the

[12] The ability of a computing process to be used in multiple contexts or by many people.

democracy walks. This labor could also be mobilized towards gaining support for a cause, especially donations to the MayDay superPAC to support its social media campaign and to pay contractual staff. Aside from these actual needs, some pitches were actually creative, including TownHall, which is the subject of this chapter. The team coded a draft of their API in the 48 hours they had at the hackathon and left it at that.

Maybe an even greater indicator that Hack 4 Democracy participants were part of an existing community of practice was in their ability to chronically cross-reference what other groups were doing. The TownHall and Pledge Tree teams both commented on Gabe's hack for rating politicians, VoteSquared.org, which was developed prior to the hackathon. It should be noted that no other hack was built prior to the event.

Table 9 lists all the hacks demoed at Hack 4 Democracy – not the ones that were pitched, but those that were demoed at the end of the event.

Table 9 – List of Hacks Demoed at Hack 4 Democracy Boston

Team Name	**Value Proposition**	**Categorical Innovation**
Data Liberation Project	Made a web crawler for downloading public data and process pdf data into machine-readable data. OCR handwritten answers.	Data exchange
Battle of the Story for a Better Democracy	Power analysis of narratives that inhibit change: creation of new narratives	Storytelling
TownHall Multi-Scale Decision-Making Tool	Tool for collective organizing enabling groups/organizations to form coalitions, make decisions, and work together.	Networking
Radio Republic Regained	A think tank and media network dedicated to consolidating news covering money in politics.	News aggregation
Anti-Corruption University	Experiment in models for consensual communication	Political Education
VoteSquared.org	The Social Platform for Rating Politicians	Voter Analytics
Pledge Tree	Track which social hubs are making	Voter Analytics

	progress creating a movement with a "pledge tree."	
Politricks	A hub for media that supports the movement for anti-corruption political reform.	News aggregation

The trends in these hacks indicated that voter analytics (two hacks out of eight) and the aggregation of news covering a single issue (two hacks out of eight) were highly desirable for some of the nonprofit sponsors, notably Democracy Matters, whose Max Stahl had pitched the Pledge Tree hack, and Open Democracy, whose team worked on the 50,000 voices logistics portal for Granny D's bi-yearly walks to raise awareness on Big Money in politics. Professor Lessig had pitched the idea behind the Anti-Corruption University hack, upvoted by a judge-less audience as one of the viable projects built at the hackathon.

From field observations with the nonprofit organizations that were present at Hack 4 Democracy Boston, I noted the nature of these hacks as lacking the element of competition that polarized the distribution of authority at healthcare and journalism hackathons. In this case, civic hacks had great use-value for the communities that supported their development at the hackathons. Although the participants did not know each other prior to H4D, they knew how they related to each other, as nodes in the actor-network that made up the movement for campaign finance reform in Boston. An egalitarian ethos of production subsequently dominated their interactions.

Table 10 – Distribution of Expertise for the Principal Group Studied in Chapter 2

Ralph	PhD Candidate in Applied Math at Florida State and founder of UnKochmyCampus
Jeremy	Middle-aged serial entrepreneur with degrees in physics and computer science and a long career in consulting for startups in the Boston area. Got his start at Apple and IBM in the Bay area.
Devers	Freelance frontend web developer for the Represent.us Boston. BA in Computer Science from UMass-Amherst. Now a senior software engineer at an IT startup in Boston.

Dan Porter	Freelance backend web developer later contracted by MayDay Labs; now lead developer at PricewaterhouseCooper. BA Environmental Studies, Florida State.

Table 10 lists the names of Townhall hackers, along with their school and employment histories. The Townhall team was made up of four and some rather precarious workers: mathematician Ralph, who was finishing a PhD; self-taught programmer and freelancer Dan Porter; Devers, a freelance developer for Represent.us; and Jeremy, an engineer and consultant to various technology companies in the Boston area. Dan had traveled to Boston on his boat looking for work at the same time as his best friend Ralph was moving there for a postdoctoral fellowship at MIT. The friendship ties between Ralph and Dan Porter constituted an anomaly among my informants, who for the most part did not know each other prior to the hackathon. This contributed to the light mood of the collaboration, whereas Dan brought to the team a highly sought-after skillset. We will see from their virtuosity that they each gained their authority through their intellectual (primarily linguistic) labor, which produced ideas, codes, texts, and images (Negri and Hardt 2000).

The relationship between these volunteers and one of three organizations at this hackathon – Represent.us, was closely tied to contractual work opportunities after the hackathon. Represent.us was one of the organizations to put up the event and several representatives in addition to the Harvard and MIT students were available onsite to hire immediate technical help on ongoing projects. At the time of writing, one of the team's members (Dan) was contracted by MayDay Labs to work on a separate project I tracked. Devers continued to work at Represent.us for a total of three years before joining several startups. But while Dan continued to work contractually for another two years, both have by 2016 secured permanent "senior" positions at technology companies. The exception to the freelance mode was Jeremy, who had been hired full-time by a major healthcare IT company the summer prior the hackathon. Even if the

remaining three members aspired to a job, they were committed to their activism (for the exception of Dan). Activism is a field of work that frequently needs a public, and they taught each other about the kinds of work they had done in that space.

In Example 1, Jeremy is shown to possess a certain aptitude for creative programming. The following two-minute segment involves the two programmers discussing the technical conditions for building and integrating an authentication service with user login information.

Example 1

1	Jeremy:	I don't know. I was just exploring the idea of extending it to users… I
2		guess what happened, what you're planning to do when you log in, it will
3		be part of the context. So you'll keep track, your connection will have a
4		property
5	Devers:	//Um-huh …
6	Jeremy:	That'll be user ID, you'll never have to specify that again. In other words,
7		it's a hidden parameter of the RESTful API.
8	Devers:	Exactly.
9	Jeremy:	But it's part of the context, the ***connection*** context.
10	Devers:	Um-huh. (.) Passing around a HTTP header is specifically the way it's
11		being done. And so there are couple of paths you can take to get to this
12		opposition header. I mean, in my head, we could… there would be a
13		couple of paths. We could, log in through Facebook, and in with the
14		password, Google or whatever.
15		Ultimately, whenever they've been verified, authenticated, we'll give
16		them an authorization token
17	Jeremy:	Well, every HTTP request, does that include ***that*** as a header?
18	Devers:	So, yes. That's how we identify that this is a valid request.
19	Jeremy:	'Cause, another way to do it, which I've used, on the server side, you've
20		got connection parameters that's stored with the connection, so as long as
21		you've got an open TCP connection, it keeps whoever the user is. I did
22		that with Node.js
23	Devers:	Okay, that's for asynchronous stuff
24	Jeremy:	Uhm?
25	Devers:	That sounds like it.

In lines 1-9, Jeremy instructed Devers on the possibility of storing a connection context, known to have the property (line 4) of user ID (line 6). In lines 6-7, he explained that user ID would be a hidden parameter if they used the REST framework to build the user login. The

REST framework helped programmers format their instructions. Devers interjected in lines 9-15 to elaborate on Jeremy's proposition, explaining that each instance (specific realizations of a class of objects) would return an access token after the server authorized a user. The way it could be done, and he clarified that there were several (lines 12-14), was to query data via an HTTP header. That is, they could use two verb methods in a Hyper-Text Transfer Protocol (HTTP): *post* to send data to the server, and *get* to retrieve the information, to display it onto the client browser. Mediating interactions between the client interface and server was the access token.

```
token = AccessToken.objects.create(user=user, client=cl,
expires=datetime.date(year=2015, month=1, day=2)
)
if self.request.accepted_renderer.format == 'json':
response = Response({'access_token': token.token})
response.status_code = status.HTTP_201_CREATED
return response
login(request, user)
return redirect('/users/' + str(user.id))
```

Figure 11 – Code to Create Access Tokens
Source: TownHall Github's View Module

The Wiki page in the View module lists all the verb methods – called "specifications" in the world of programmers, for pulling group and user data from the TownHall API. The first point of contention between the two programmers regarded the location of such permissions, visible or hidden. In line 17, Jeremy asked Devers if the authorization taken was part of the header (a HTTP header takes the first line of the request to specify the endpoint of the resource in order to fetch it from the backend database). Jeremy's expertise was not yet called into question; what this transcript show us instead is that we can have two programmers with different practices, or paths to solving the verification problem they raise. This heterogeneity itself led to a brief teaching moment (line 18 "yes, we identify that as a valid request"), whereby Jeremy asked where to look for some information ("does that include that as a header?"). Devers

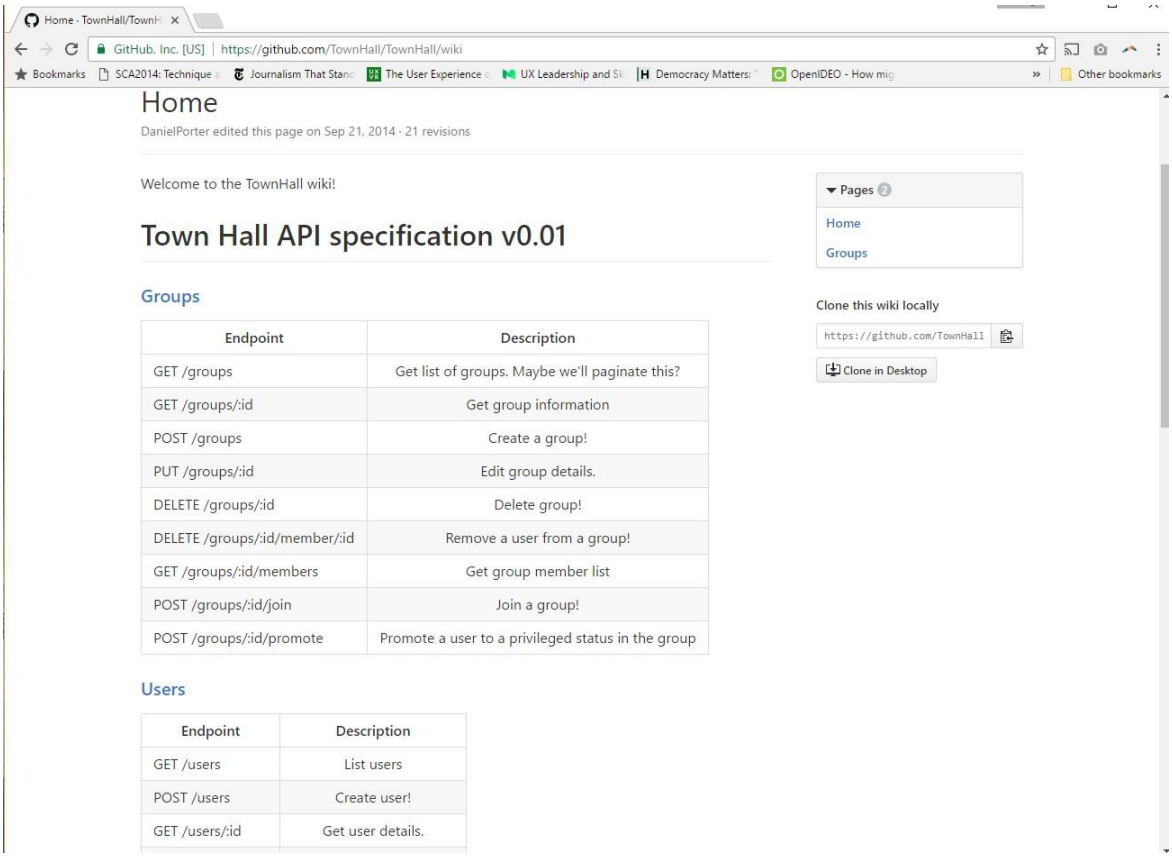

Figure 12 – Townhall API Specifications or Syntax (or Rules)
Source: TownHall Github Repo

confirmed the latter's hypothesis on the location of said resource, adding that the machine would look for it there as part of the validation procedure that authenticated users.

A second teaching moment occurred in lines 19-22. By then, Jeremy had proposed a second option, recalling the possibility of keeping an open TCP connection in Node.js (lines 19-20). Node.js was a framework for programming applications in Javascript – the language of the web. Devers' intuition, however, was that keeping an open TCP connection, which would require that users keep their status as 'active' – an unfeasible scenario these days, was not a legitimate practice. The latter immediately dismissed Jeremy's suggestion by clarifying that open connections were only useful in asynchronous communications (line 23 "that's asynchronous stuff"). Jeremy was not able to carry the conversation further once that distinction had been

made.

An interesting point can be made from Devers' closing of the debate at this stage. It would seem that what drove the members of the Townhall team in this stage of the brainstorm was a meta-level commentary on specific coding practices. It partially shows their virtuosity as hackers as fragmented subjectivity. Jeremy pressed for more context by expressing confusion (line 22 "hmm?") but all he got from Devers was "that sounds like it [asynchronous stuff]" (line 23). This closing statement suggests that Devers was not altogether sure that the practice of keeping an open TCP connection is a property of asynchronous communications. In line 25, Devers smoothed his statement over by voicing a weak conviction that it is a bad idea in practice.

There is a consensus among programmers at hackathons that if one's teammates could not elaborate on each other's propositions, that one might lose the support/labor of teammates by stubbornly pursuing an avenue where they could not follow him. The following section now demonstrates how the Townhall team eventually reconciled their differences in programming practices.

Managing cooperation through humor

Dan eventually chimed in with a proposition that neither Devers nor Jimmy could refute. The scene in Example 2 features the orchestration of a collective joke to diffuse the earlier disagreement between Devers and Jeremy. By then, Jeremy was reiterating what he had attempted to explain to Devers (line 1, "it's [the login credentials] part of the connection") when he caught sight of Dan successfully accessing some classes of users i.e. groupsID, groupsIDmembers (line 1, "you got something?"). Dan waited for his machine to process all the information he was retrieving from an API he was testing, and declared that "it [his machine] just returned a history [or "some groups," line 2]. Dan suggested doing a mock API (line 2),

which would give them full control of the environment he was inspecting. The groups they would receive from this early experiment would only model the class of credentials they wanted to transfer between client browsers and the server. It was, altogether, a better use of their time.

Example 2

1	Jeremy:	It's part of the connection. Some groups… You've got something?
2	Dan:	Yes, some groups. We could do a mock API
3	Devers:	I'm sorry, if you wanted to be a shmuck or something, for talking so
4		much.
5	Ralph:	Yeah! Me too, I've got no other option.
6	Dan:	It's your job, it's your ONLY job!
7	Ralph:	It's my job to be a shmuck.
8	Devers:	I was just describing my skillset to her, I guess yours is, it's captivating
9		shmuck.
10	Audrey:	I should write that down.
11	Ralph:	I should write that down too.
12	Jeremy:	That's derogatory.
13	Everyone:	*((laughs))*

Despite not knowing each other, Devers responded by playfully calling Dan "a schmuck or something" (line 3) for disqualifying his lengthy discussion of the RESTful API's problem (line 4, "for talking so much) by suggesting, quite rightly, that a mock API could do the job (line 2, "we could do a mock API"). This joke was an attempt to mitigate one's embarrassment and restore Dan as an equal collaborator, despite his stronger knowledge of the problem. The team repeatedly used humor to balance each other's authority. Despite not knowing each other prior to the event, Devers' joke legitimized Dan's proposition in a way that rattled everyone on the team into helping Devers amplify his message and build consensus. When Ralph ventured that he might be a schmuck too (line 5, "Me too, I've got no other option!"), his best friend Dan interjected by confirming that it might be his "ONLY job" (line 6). While the two best friends might have been laughing off their precarious work conditions as PhD student and freelance developer, Devers' joke also served the other purpose of building his affinity with Dan. Devers was implying that he should have thought of creating a mock API himself. For my teams at the

Athena Health hackathon in Chapter 4, it was standard practice to use mock APIs to test a proof of concept.

Technical translations

RESTful APIs serve all the current practices for content negotiation on the web, and already include some instructions on how to apply them. Regardless of how voluntary the standards in hypertext protocols are from the onset, Devers and Jeremy structured their debate around them, to eventually build off an existing library[13], called *oauth* in this case, to allow for interoperability in the long run. The idea was to have the machine automate the verification of users for any given number of contexts, which the developers would specify. The context could be fabricated remotely!

Ralph envisioned a platform where activists could all go to do work; the API was only one process in the hack. Ralph was not a programmer, and his notional solution to the problems they anticipated in building the platform left a big gap in the developer's understanding of how the machine would perform verifications without a human. The programmers debated how to execute Ralph's hack on a more technical level in Example 3.

Example 3

1	Jeremy:	And if we had *this*, this provider way, really, of giving true online
2		petitions, being able to get people by voting in your local organization
3		that we trust, it could accumulate and get a lot of people signed up
4	Ralph:	Right, because it's multi-scale trust network. You create a supergroup
5		because you have ***consented*** to being affiliated with the organization
6		and so it's no longer "who is this?" and "what is this e-mail?" It's part of
7		the thing that you've been working on. And more as well, a petition that
8		travels like that, is acknowledgment that people will support this idea,
9		but now that there is, if there are work groups built around that
10		structure, not only can you all agree on this petition, but you can then

[13] In programming, a library is a collection of precompiled routines that aprogram can use.

11		in the supergroup decide what to do about it once you've agreed that
12		this is a thing that support. Now let's get a proposal, how do we act?
13	Jeremy:	You can automatically get support because this petition is supported
14		by this group, this group, this group…
15	Ralph:	That's exactly the idea of this large-scale alliance building. Because that
16		support, like, you know, an organization I belong to could be doing
17		some-thing that, potentially, could be supported by hundreds of groups
18		across the country, but how do you ***verify*** that? How do you get them
19		to literally sign on, how do we get them to like, be official allies? And
20		create a structure of coalitions such as that?
21	Jeremy:	The structure you have here for having agendas and votes will have
22		members of the group decide, "do we support this petition or a vote of
23		the organization.
24	Ralph:	So they could abstain? Of they could. If you get a majority, it says "this
25		organization support this petition." (.) I think that's powerful.
26	Devers:	I think, I think, I hadn't thought about it in terms of petitions. It's
27		Interesting, I don't think about petitions because in my world, they
28		seem to be a source of power, but maybe they are... I think we want
29		more effective power.
30	Jeremy:	Petitions are usually good for building a list. They don't usually count as
31		much.
30	Ralph:	So how do you act on people pledging their support? So in this case, it's
31		not "cool, you've got a big listserv, you can e-mail people actual alerts,
32		but-
33	Jeremy:	They don't care, the petition isn't the point, they only want a name
34		*((laughs))*
35	Ralph:	But whereas, if people are consenting, "yes, I want to work on stuff," and
36		if they go to a place to work, it's not an e-mail coming into their private
37		box, it's where they go to do work. So if this is where all of this is taking
38		place and they have consented every step of the way, that's ***powerful***, (.)

In response to Jeremy's observation that trust in a local organization was a strong factor in accumulating support on online petitions, Ralph reframed the programmer's earlier emphasis on the technical conditions of support and verification as a social challenge (line 6 "who is this? What is this e-mail?"). Jeremy felt civic technology – mostly information technology that enhances citizen communications, and improves government infrastructure for the public good –

could account, mediate, and transform some of this symbolic work by envisioning a platform that could "learn" the pattern of support that a user disbursed between various political advocacy groups, such that one "could automatically get support because this petition that is supported by this group, this group, this group..." (lines 13-14), or each node in a network. Ralph subsequently elaborated on Jeremy's proposition by identifying the user action that they could qualify as support: if one consented to affiliation, the implication was that his or her statement of support for the work of a group could "travel" to the petition (lines 7-8). In other words, their joint labor in which they imagined the possibilities for accumulating digital labor on a platform was building from the premise that user actions were extensible and could be replicated to make multiple statements (Latour 2005, 54). An activist's participation in digital labor and now immaterial labor could now challenge various assumptions of collective trust.

In lines 26-39, the discussion moved to new imagined forms of consent in the digital age. How does one go about recruiting digital labor for the production of otherwise traditional civic forms of life (i.e. marches, petitions)? In lines 26-29, Devers voiced his reticence to embrace petitions as an effective source of power, and challenged the very assumption underlying any theory of social movement in the United States. Jeremy echoed this perspective in line 29, by agreeing with Devers that "they don't amount to much," yet insisting that petitions could help any program build a list. Ralph nevertheless proved to be effective in embracing deviations to his path. In line 30, the academic type reframed the challenge as an opportunity for the developers: "How do you act on someone pledging their support?" Ralph walked his team through this new direction by brainstorming the location of the feedback mechanism for a pledge. "You can e-mail people actual alerts but" (line 31), discounting the short attention span of e-mail readers, he brought the team to a new possibility "if they go to a place to work" (line 38), this action and that

attention as work that makes a difference in for activists. Ralph speculated with great enthusiasm that directing traffic to the TownHall website was the best way to establish whether one's consent was secure. Labor on their platform, he reckoned, could be treated as intentional, or "consensual."

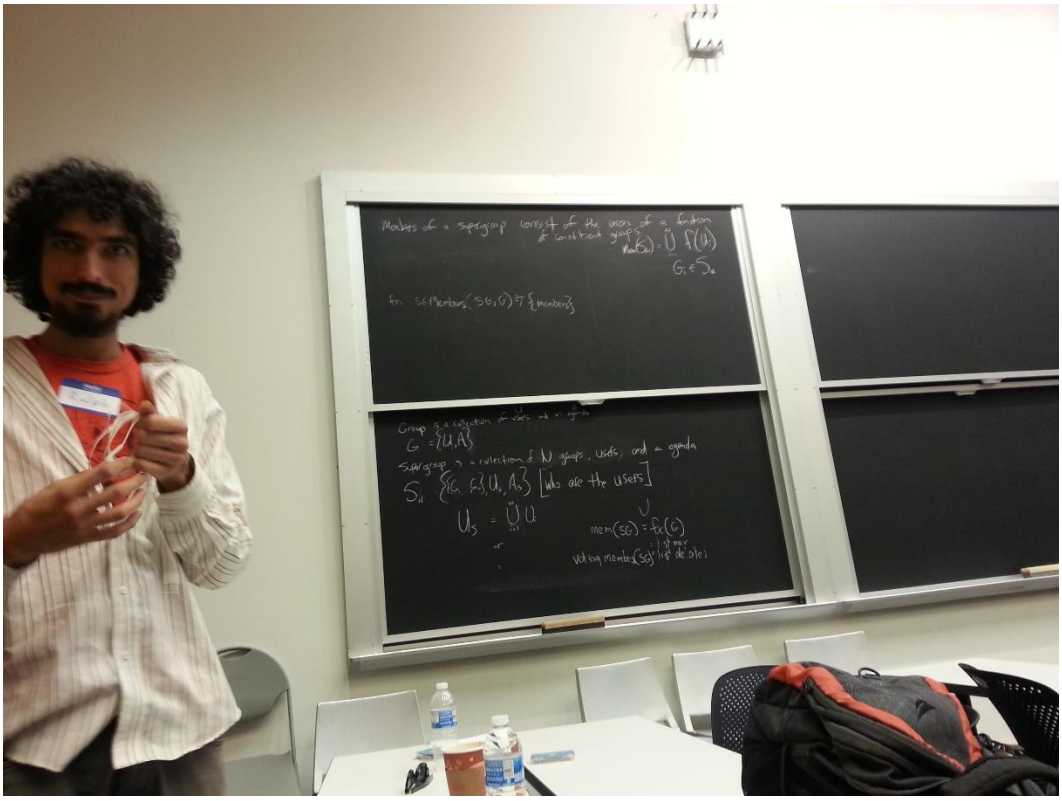

Figure 13 – Ralph's Equation

Figure 13 shows Ralph's equation on which the team's notion of consent was fabricated mathematically. Put simply, his knowledge of the theory of complex systems encouraged Ralph and the team to think nodes in a network as being modified recursively. At the same time, this form of de-individuation is what makes post-Fordist labor organization possible; the team only had to reach consensus on which feature to automate. Ralph, Jeremy, Devers and Dan were engaged in a critique of current forms of political action, and transposed this meta-knowledge onto The Game of Life, a math-rendered gamification of social life. Despite its intricate detail, The Game of Life communicated, albeit in geeky language, how programs could make their

work more efficient and transformative at the same time.

Example 4

1	Ralph:	That is the thrust of uh, chaos theory and complex systems, is that you
2		create a simple rule, and that rule when applied, that can create very
3		complex algorithms
4	Jeremy:	Like a math problem *((laughs))*
5	Ralph:	Right. Like a cellular automaton. They're, they're the same thing.
6		Amanda Brooke said, uhm, it's generated by a given function, and you
7		iterate it and you have a scheme for what's out of the set.
8	Devers:	By the way, did you see that Game of Life emulator that, uh, runs the
9		Game of Life, it's absolutely incredible. You know, John Con… John
10	Ralph:	John Conways[14], Game of Life, yeah.
11	Devers:	Someone came up with a system that actually runs, that runs on a higher
12		-level of Game of Life. Each cell, you know, the giant cell consisting of all
13		that stuff interacts with other giant cells, and it makes things literally in
14		the dyad at a higher scale, it's like a fractal, come it's really cool, uhm
15		you can, you can, you know it shows like zooming out, it shows the little
16		individuals, gliders and stuff zooming out. It's like another Game of Life.
17		*((laughs))*
18	Ralph:	That's ridiculous.
19	Everyone:	*((laughs))*
20	Ralph:	Have you seen a Turing Machine? So, so… I forget who it was, it was
21		probably here, because that's where stuff happens
22	Everyone:	*((laughs))*
23	Ralph:	Somewhere in Cambridge, some people demonstrated that you could
24		build a Turing Machine out of The Game of Life, uhm, a universal, a
25		universal uh, computing engine,
26	Devers:	Wow, that's
27	Jeremy:	WOW
28	Ralph:	just strictly made out of gliders, and like, making basically, circuitry out
29		of gliders, and creating a universal computing machine
30	Devers:	That is… sooo…
31	Ralph:	F*ck… out of The Game of Life
32	Devers:	That's part of what makes computers in Minecraft accurate
33	Ralph:	Yeah! Actually

Jeremy and Ralph shared a common background in experimental physics, and showed some excitement at the time Ralph wrote the equation on the board. For a while, the team discussed the multiple scales of collaborative decision-making on digital platforms, suddenly

[14] See John Conway's paper here: http://rendell-attic.org/gol/tm.htm

veering towards a passionate discussion of different hacks. The affinity they shared from such eclectic references to popular culture led them to *geek* out (Ito et al. 2009). Geeking out is a combination of intellectual and affective labor, whereby youths and adults alike demonstrate their expertise and passion for a subject. Eventually, experts can make money from their skills, but what I would like to emphasize here is that geeking out – despite its temporary diversions – facilitated Townhall's creative problem-solving process.

From the moment the Townhall team started to geek out, the boundaries between work and play increasingly got increasingly blurred. Were they participating at the hackathon out of a strong sense of civic duty, or did they work on this problem because they had fun together? The answer was in their care for craftsmanship. As they discussed memory charges, the Townhall team bonded over Ralph's equation, the Game of Life, and Minecraft. It seemed surreal to me, because the imagery – fractal and dyads (line 14) and gliders (lines 16 and 29) – preceded augmented reality; these guys were talking about making all these connections right in front of me to indirectly talk shop. The transcript shows Ralph sharing his deeper knowledge of the math behind some of the hacks that they discussed – the Game of Life being in its essence mechanically enhanced by a problem that a math whiz in Cambridge had cleverly solved (line 23). Devers was able to lend an audience to elaborate on Ralph's thinking and let him exert himself. Despite misjudging the memory capacity of the Minecraft game later on, Devers was a proficient adversary in this workshop session, geeking out and amplifying Ralph's vision. Together, they participated not because they felt they belonged to the hackathon space, for they had spent some time discussing whether they were actually making a contribution, but because their skill level was the same, they continually tried to prove their abilities to each other. This virtuous conduct created the own contours of their participation at the event.

Positioning oneself towards immaterial labor

They never finished hacking. The team took notes on a Hackpad page, and created a Github repository to share the code they had already written for anyone to edit, improve, or build upon.[15] The following screenshot of the "Next Steps" they wrote on their Hackpad confirmed the kinds of immaterial labor – feedback mechanisms – needed to valorize their code (Figure 14). In the bottom of the page, the team wrote that they needed "good API documentation"; "a way to easily learn from the system"; and to know "current practices for open-source projects."

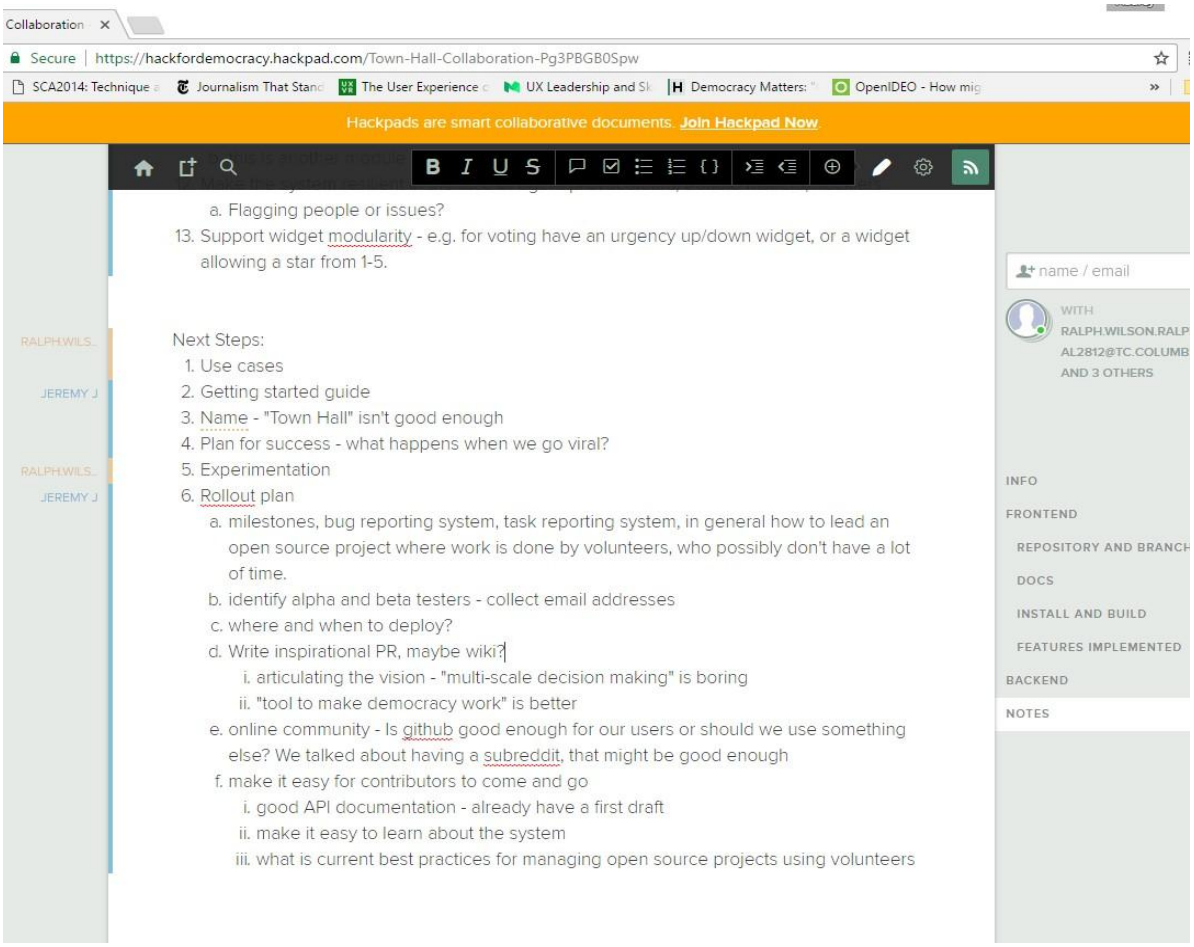

Figure 14 – Next Steps for Coding TownHall
Source: Hack 4 Democracy Hackpad

[15] It should be noted that the practice of sharing code and documentation does not accomplish the same goal as sharing content on social media – that of circulating messages and images – but, it does rely on a spectator to valorize its content whether it is judging the potential for investments in the code as fulfilling a need in industry. Code depends on the scarce attention other developers with the skills to manipulate its symbolic-analytic structures (Negri and Hardt 2000) to animate it and give it a purpose outside its immediate production.

Daniel Porter made a few formatting changes a few weeks later, before abandoning the project for lack of ownership and the necessary labor-power to keep working on it. Statement 6a is particularly revealing of the changing nature of that work post-hackathon: the first suggestion to the rollout plan and the primary task that the team gave itself was to recruit someone who could help develop milestones, fix bugs in the reporting system, and dump all the content from discussion threads onto the Town Hall platform to encourage leadership on future open source projects by increasing its visibility. These practices highlight the team's desire to create better channels of communication, which support Virno's (1996) exploration of the intellectual and communicative aspects of the masses as the new relations in the production of consent in digital activism. The increasing use of automation, according to Virno, would have the effect of de-skilling workers, but Lazzarato offers a stronger understanding of these shifts in value capture. In the case for H4D3 participants, labor processes have placed new value on certain skills of cybernetic control without de-emphasizing existing work practices. Still, in the process of becoming subjects of communication, the Townhall members have come to desire a place where anyone could see the work that others were already doing. More specifically, they re-appropriated their immaterial labor to fit their own needs as activists. It is their virtuous conduct, as the articulation of their subjectivities that create different end-goals in the course of their collaboration. Of the many tools they considered appropriate for the task, the team expressed a preference for Reddit to redirect activists' digital labor to their platform. Verification was also a classic coding problem that had been applied to the labor of the activists, but it was challenged in Example 3.

Chapter 2 Conclusion

This chapter discussed the Townhall collaboration. The team exhibited some of the most ludic aspects of play labor, which I summarize now. LIBOR Alt + Repair and the Campaign Con – the other two civic hacks – will be discussed in Chapters 5 and 6, respectively. For now, the Townhall team is enough to give us an idea of the type of work and play that civic hackers do.

As a consequence of the ability to sustain their attention on multiple related issues, the Townhall team's play labor had diversified. The transcript in Example 3 shows the group of four activists building a procedural awareness of hidden biases in the design of their supergroup system. Example 1 described the debate between two programmers on the team, which led to a teaching moment in which they deliberated over several paths they could take to verify users. The problem was first a technical one, but it was resolved using their subjective experience of new forms of immaterial labor on the web, which had complicated the work of activists, who were puzzled by new abilities to communicate with their campaign supporters. Examples 2 and 4 illustrated how team members managed each other's cooperation through humor and geeking out. Given that only Ralph and Dan knew each other, their virtuosity helped these strangers reconstruct each other as accountable subjects during the hackathon. Their performance did not gain recognition from peers at the hackathon, but it did raise a few roaring laughs, which grabbed the audience's attention on some perennial issues in digital activism.

The examples in this chapter showed the experiences that Ralph, Brad and Jimmy brought the technical and conceptual problems they encountered in building their multi-scale decision tool. The main problem was however industry-specific: consent was by the order of the US code (18 U.S.C. § 597, 1999) illegal to buy, a fact they duly addressed in Example 3. Knowledge of such an issue was important in assessing the merit of the hack. Consent reflected a

much stronger currency for activists, who were always in need for more free labor and needed it to keep it ideologically removed from monetary concerns. In terms of the work practices they brought to the hackathon, Ralph's experience with the Occupy movement had left him desiring a tool that would let the protesters systematically present demands. During his interview on November 14, 2014, Ralph explained that in 2012 each group of supporters brought fifty proposals back to their cities to ratify them accordingly. At the time, there had been no mechanism to keep track of their progress where the work could have been automated. The movement, as a result, suffered from underfacilitation, as he put it. Hundreds of people were writing down different lists of demands, and no one knew how to pool their collective labor to publicize an issue they cared about. Ralph's experience framed the learning curve for participants in his team; the contexts needed to be much more granular than he had anticipated to organize the team's technical workflow. Their debates over which path to follow illustrated the forms of digital labor that required Ralph to develop managerial skills for coding.

The case study of Townhall hence highlighted issues in the practice of US-based digital activism, such as signing petitions and ratifying proposals. In 2014, a number of civic startups had emerged on the scene to incite new desires for technology among activists. But it was the team's play labor (geeking out, humor and a general passion for math) that inspired the creativity of the hack. Participation did not only require skills for manipulating and controlling user information but playfulness and skills at managing social relations of trust. Civic hackathons drew on the virtuous capital of their participants to support the work that they were already doing, with an eye to their special sensibilities as activists in a particular space, which they asserted to make sense of the risks and rewards for technology-enabled political action.

Chapter 3 – Mistranslation, Role Reversal and Pivoting in Healthcare

Healthcare hackathons mostly employed a catalytic model, seeding innovation with an articulated challenge and fostering a competitive work environment, with a minimum of skill or idea qualification to participate. Competitive hackathons like the MIT Grand Hack and the Athena Health MDP series sought the continuity of ideas; they aimed to demonstrate the utility of new technologies or approaches that were novel and tractable (Drouhard et al. 2017, 2). In the case of the Grand Hack, participants were encouraged to launch startups at the end of 54 hours, while Athena Health MDP participants were invested in ideas that they could sell on the company's digital marketplace.

Three healthcare hackathon teams – Sirona Care, Reciprocall and The Cuff Guys – unstable and highly variable in their play labor, are the subjects of this chapter. The kinds of play labor we will see are: pivoting, the reversal of roles and mistranslation. The reversal of certain roles was the outcome of having a larger team; healthcare hackathons commonly produced teams of seven to nine people to account for all the skills required to build hardware. With the exception of The Cuff Guys, a team of four (and later three when the developer dropped out), the teams averaged a size of nine members and therefore had to find creative ways to manage each other's cooperation. Sirona Care used a very rigid division of labor, while Reciprocall did not use any structure to manage the contributions of its members.

The MIT Grand Hack, April 24-26, 2015

I start by describing the ecological histories of digital labor at two healthcare hackathons. Then, I discuss the kinds of experiences that teams brought to the hacking process through the case studies of Sirona Care (MIT Grand Hack) and Reciprocall (Athena Health More Disruption Please!). Finally, I compare their organization of work and its impact on the value-add of

different technologies.

Upon founding MIT's Hacking Medicine club, Zen Chu had announced that the time was ripe for healthcare innovation. According to PricewaterhouseCoopers (PwC), this seemed to be the case. PwC reported the following breakdown of deal sizes, totaling 193 deals for $2.2 billion:

- Biotech human – $1.35 billion
- Medical device therapeutics – $258 million
- Pharmaceuticals – $212 million
- Medical diagnostics – $107 million
- Medical/health products – $104 million
- Biotech research – $73 million
- Biotech animal – $35 million
- Biotech industrial – $27 million
- Biosensors – $20 million
- Biotech equipment – $11 million

By "the time was ripe," (mentor's workshop, April 24, 2017) Zen Chu must have alluded to the fact that investment in medtech and the life sciences was waning. More specifically, in 2015, investments in the medical device category had dropped by 30% and in biotech by 4% from the previous year (PwC 2015). This meant that more investments would be made, instead, in health IT and telehealth.

To gain momentum, 56 healthcare hackathons were held in 17 countries outside the United States between 2010 and 2014. Two organizations have helped aid this expansion: Health 2.0 hosted code-a-thons in the Netherlands and China in 2012, and the Canadian organization Hacking Health held hackathons in Switzerland, Hungary, and Sweden in 2014. Hackathons were also playing a growing role in the developing world. In July 2014, the Massachusetts General Hospital's Consortium for Affordable Medical Technologies (CAMTech) co-organized a hackathon in India called the Jugaadathon.

The MIT Grand Hack had a 90% admission rate; the 10% rejected were largely composed of MIT students to ensure that the club met its mission of diversity at the event. The MIT

Hacking Medicine club organizers of the Grand Hack had empathized with the long essays applicants had submitted to gain admission to the event; the applicant had to answer two questions "what is your idea?" and "what expertise do you think might help you at the hackathon?" to minimally qualify for the Grand Hack. Adhering to the idea qualification criterion helped ensure people had the dispositions to succeed during the event (Lina Colucci, organizer's meeting, March 8, 2015). Among those accepted, one organizer shared the following statistics: 30% came from Boston; RSVPs counted nine countries (including Qatar and Uganda who came from CAMTech) and eighteen states (including Rhode Island, New York City). The biggest group (221 people) counted ages 18 to 24. Fifteen people were 55 years old or older; 35% were female and 39% were students. Although the organizers initially feared to contact students from Harvard, who had in the past been notoriously "flaky," 8% of those registered for the Grand Hack were from Harvard, 10% were MIT students, and a lot were from Yale (Katie Chung, organizer's meeting, March 8, 2017), indicating that prestige was at stake in this competition. But as Aaron Lifshin (interview, October 28, 2014) once commented, students had a flexible class schedule and could afford to spend a weekend at a hackathon. A lot had also stated on their application that they had founded a company, won a prize at another hackathon or were looking to recruit at the event. The few angel investors who signed up were interested in knowing the trends in healthcare innovation.

The startup weekend-like hackathon was divided into four tracks: Wearables, Primary Care, Global Health, and Telehealth. The fourth track responded to VC demand for innovations in telehealth. With these tracks, the Grand Hack was the most organized hackathon after Hacking Journalism NYC in Chapter Four, for which the organizers assigned teams. Organizers encouraged participants to select a track in advance, but did not limit them to their first choice.

They also designed the event to have business and legal mentors available around the clock, and readily gave individuals advice from their years of organizing and seeing what made teams successful at startup weekends. Despite all their good intentions, the sheer size of the hackathon, which was attended by over 400 participants, had the most chaotic feel. The open space had four rooms with roundtables but no room for private talk. This made for a very loud event, with organizers routinely shouting reminders on microphones. To solve for some of this confusion, Zen Chu had convened all mentors before hacking started, and explained that the energy of the hackathon was important. Add to this "controlled chaos" a dozen of vendors, and the Grand Hack started to feel more like a capitalist intervention to the entrepreneurs. Tarek, a repeat participant at healthcare hackathons, expressed his amazement for never having seen that many vendors (Tarek el Shayal, pers. comm. April 24, 2015). Emily from the Ultrasound Training team at The Grand Hack had commented that the tracks had made it more difficult to self-organize, but that she nevertheless understood why the orgaziners designed them: to manage the crowds at the event.

Next, the organizers selected speakers based on their experience in the innovation space; especially, they welcomed previous hackathon participants who had turned their hack into a product after the event. With such a background, the organizers claimed speakers were more adept at finding good entry points for the technical people at the event. Dr. Ken Moritsugu, Medical director of CAMTech (Consortium for Affordable Medical Technologies) discussed how pedestrian safety was a big issue in megacities of resource-constrained countries, where roads are designed by the private sector. Anders Wold from General Electric Ultrasound remarked that physicians were not trained in primary care diagnostics for complex cases. Other speakers represented companies that wanted to learn from participants how they could innovate

their technology to enter new markets. GE was well-known for portable ultrasound technology, a low-cost, low-radiation ultrasound device that had yet to make it into primary care practice. Wold talked extensively about the ecosystem of GE and framed many opportunities to contribute to it. For instance, most ultrasound technology operated in single mode and had yet to be connected to the cloud. Because the portable ultrasound technology was offered at the hackathon, Wold invited participants to contribute to its business viability, not in terms of GE per se but for the future of ultrasound practice. Healthcare had always been perceived as an industry slow to keep up with new technology. Now that Americans were eating healthier and getting insured, healthcare, according to these speakers, was becoming one of the largest consumer markets.

Maulik Majmudar, the Associate Director at the Massachusetts General Hospital's Healthcare Transformational Lab, took caution in separating himself from the business of Microsoft before speaking at length about the market for wearables. Citing Google Ventures' new report, the cardiologist forecasted a six-billion-dollar industry that could potentially alleviate rising healthcare costs, and take advantage of the computing power of smartphones. It is now easy for patients to look up information online. By connecting wearables to phones, one could improve self- management of chronic diseases. Mr. Majmudar, however, observed a significant drop in the use of fitness wearables like Fitbit after six to eight months. Research he cited conducted by Rockhealth indicated that most users had very low social and medical complexity; the problem, according to the director, was not with the limited health functionality of wearables, but that they needed proper calibration to respond to the special needs of users with chronic diseases. He suggested integrating data streams; new analytic platforms to generate insights; some principle around behavior change; and clinically meaningful pain points for users.

The talks on Friday night highlighted the problems with smartphones and wearables e.g. devices such as smart watches and fitness trackers, which made it possible to collect and learn new information about a user, but lacked the personalized marketing insights. Asking the crowd to develop new solutions around this problem, pitches voice participant ideologies surrounding cloud-based computing and personalization. The lists of hacks in Table 11 below further confirm the value that hackers at the Grand Hack assigned to these vehicles of business innovation.

Table 11 – Sample Hacks for the Wearable and Telehealth Tracks

Team Name	***Summary of the pitch***	***Business innovation***
TO-U	Use portable ultrasound for a permanent female contraception procedure (CAMTech GH Runner, $500)	Enhanced imaging
Pillar	A smart bracelet birth control dispenser that integrates with Microsoft Band and a mobile app with intelligence algorithm for non-compliance recovery instructions (Microsoft Wearables Track First Prize ($1000)	Medication Adherence
Diabeat-IT	Activity tracker for people with diabetes (Wearables Track Runner-Up, $500)	Fitness Coaching
Smart Wash	A smart watch-based app to help health staff adhere to hand-washing protocol (Wearables Track Runner-Up $500)	Compliance
Food Genie	Monitor diet by collecting intelligence on restaurant choice (Jawbone prize, UP Bands for everyone)	Personalized diet
Infinity Ultrasound	Ideal probe positioning (GE Ultrasound Prize, $1000)	Enhanced diagnosis
Simugram	Mobile app that simulates the performance of the GE Portable Ultrasound (GE Ultrasound Prize, $500)	Training, Low-cost
Ultrasound Training	Practice how to use portable ultrasound with training aid (GE Ultrasound Prize, $500)	Training, Low-cost
$1 Ultrasound	Pay-per-use (won CAMTech GH Runner Prize $500, GE Ultrasound Prize, $500)	Low-cost
Epilex	Track seizures with mobile app integrated with Myo Band (Thalmic Labs Prize, 1 Myo for everyone)	Integration with Myo

Pulsymetric	App that integrates with Jawbone, Fitbit and Microsoft Band to record heart rate and alert doctor of irregular heartbeat (Microsoft Band Prize, 1 band for everyone)	Physician Alerts
Sirona Care	Mobile app for A-Fib patients that integrates with Microsoft Band (MeErck third prize, $500, biweekly phone calls)	Medication adherence
Thermaform	3D Heating Mask for chronic pain (Merck second prize, $750, monthly phone alls)	Custom-fit
Cancer Companion	Virtual navigator to keep cancer patient on treatment plan (Merck KGaA First Prize, $1000, possible internship in Switzerland)	Virtual companion
SnapBill	Upload confusing medical bills and receive feedback within 2 hrs (Telehealth Track Runner-Up, $500)	EMR Feedback
Patient Geotracking	Automatic real-time notifications to physicians when patients is in ER (Telehealth Track Runner-Up, $500)	Workflow

Source: MIT Grand Hack Google doc

The contexts for hacking were structured around the materialities of each type of technology. The summaries of the pitches above indicate that the teams thought the best use of a wearable in medication adherence, fitness coaching, and physician alerts; portable ultrasound in enhanced diagnosis and low-cost training; and telehealth in physician feedback on electronic medical records. Corresponding to the devices available that weekend, the sixteen hacks from Table 11 tried to integrate with a wearable (seven hacks) and the GE Portable Ultrasound (five hacks). The categories for business innovation in each track addressed a problem voiced by each of the speakers representing the four tracks, such as Andrew Wolds of General Electric on the relative lack of training in diagnosing complex cases, and Mr. Majmudar of Microsoft on the need for wearables to personalize the management of chronic diseases. The buzz around Microsoft's first fitness tracker, called the Microsoft Band, and General Electric's success with portable ultrasound devices, had encouraged participants to pay close attention to new forms of digital labor around these new technologies.

Physicians were interested in getting connected to such technologies. Aditi, the physician

leader of the Sirona Care team which case study I discuss now, recently connected with me over e-mail, stating that she had learned “a lot from that weekend about articulating a problem, hearing from diverse perspectives, finding the right mix of skill sets and personalities to solve a problem, and how to create the perfect 'pitch' depending on your audience” (e-mail to author, April 10, 2017).

The case of Sirona Care

Figure 15 – Sirona Care Simulation

Sirona Care was a hack that involved an algorithm, an app, a phone and the Microsoft Band. Figure 15 shows a screenshot of the Sirona Care emulator that can be rebuilt on Android Studio. The hack was intended for the wearable to detect irregular heart rates among patients with Atrial Fibrillation and alert their doctors when the band registered an unusually high rate. This hack addressed a common problem with reliable information on a chronic condition, and

provided an alternative where any diagnostic device might run the risk of generating false positives.

Table 12 breaks down Sirona Care's division of labor in terms of the expertise and accomplishments of each member. After the second pitching session between 10 am and 12 pm on Day 1, the Grand Hack organizers had arranged for members of self-forming teams to query the room over the microphone for the kinds of expertise they needed to recruit for their hack. This strategy proved most fortuitous for Ankita and Aditi, who had pitched the same idea the night before and joined forces: four girls belonging to the same sorority at MIT (Kamilla, Joyce, Katrine and Priya) had come to the Grand Hack together and contributed most of the computer science background of the team. Their strong technical backgrounds made for a convincing selling point during the demo of their hack.

Table 12 – Sirona Care's Distribution of Expertise Contributions

Aditi	Physician resident in Internal Medicine at Harvard Medical School Doctor connections Knowledge of disease	-Visionary -Public speaking and leadership -Delegated tasks -Designed a clinical survey and circulated to resident connections
Ankita	Trained engineer Product development experience Earning an MBA at MIT Sloan	-Developed the business proposal -Pitched and wrote the demo presentation with Aditi
Kamilla	Computer Science student at MIT Knowledge of data science and visualizations	-Prototyped the Physician Web Portal -Designed interactive data visualizations
Joyce	Chemistry student at MIT Knowledge of graphic design Previously attended a healthcare hackathon	-Designed logo and slogan -Mentored Kamilla
Katrine	Computer Science student at MIT Knowledge of Java programming	-Connected to the Microsoft Band -Initially in charge of building the Android app on top of the Band

Priya	Computer Science and Electrical Engineering student at MIT Knowledge of Java programming	-Designed the app emulator with Katrine - Mentored Katrine in Java
Nabeel	Medical student at SUNY Buffalo	Researched medical laws
Jeremy	Public Relations Lead/Journalist	Designed and conducted user survey
Ning	Electrical engineer	Wrote the algorithm

By the time I sat down with Sirona Care at 2 pm, roles and commitments had already been fixed and remained the same for the duration of the hackathon. Each individual was accountable for his or her delegated task, and for that reason, any macro-level change in direction did not alter their responsibilities – only their value. Aditi interviewed fellow physicians on the pain points of managing the chosen chronic disease; Ankita developed the business proposal, while the technical tasks were distributed among the four student engineers, with strong ownership for each feature. The team regrouped every three hours. Joyce and Kamilla did not like the noisy environment of the hackathon and left the room after they had reported on their progress. Kamilla and Joyce worked on the visualizations of the physician portal that theoretically had to match the information that Ning's algorithm would produce. As the only hacker with knowledge of Microsoft's operating system, Katrine worked on connecting the Band for most of the afternoon on Day 1, assisted by Priya that night and on Day 2. The wearable was thus programmed to calculate differences in heart rhythm in relation with users' self-reported measures. The user worked to corroborate and produce the information that flowed from the device and program, and the play labor of the team was divided to account for all possible interactions between the three relations. The organization of work had the effect of buffering any disruptions in production, which was a clever way to organize their labor given that no one had hacked the Microsoft Band before. One of such delays, in the next scene,

involved Aditi and some of the programmers on the team helping each other to figure out which information to feed the wearable band.

'What is *i*?': Sirona Care's workflow based on a mistranslation

Example 1 includes a transcript detailing a problem that the group of developers working on Sirona Care, in fact, never resolved. *i* was for *index* or *iterator*, a parameter which specified the task that should be repeated or "looped." It was a common source of confusion among developers working on several platforms or in several language; every developer used *i* arbitrarily depending on the language they used to code, which had its own rules about what it should be, therefore making it more difficult for programmers to build upon each other's code. This had certainly been the case for the Anti-Corruption University team at Hack 4 Democracy, when Fred and Jacob neglected to set the conventions for merging their code up front and ended up having to recode both front and backend with Gabe supervising.

The key area of focus in this transcript was such an attempt to define *i*, and how the developers magically returned to working without knowing how each would use *i* for each piece of code they would later merge.

Example 1

1	Aditi:	Corroborated 2 user data, one user-reported of "I took my meds, I
2		did not take my meds, I forgot to take meds." To know whether one
3		actually took the meds. The algorithm, though, is almost entirely
4		based on the physiological parameters, and not the user-reported.
5	Kamilla:	*((looked up to Aditi and asked an inaudible question))*
6	Aditi:	When you say '***i***', what are you talking about?
7	Kamilla:	What information would get sent from the app? Because the
8		algorithm happens with the app, so what information is from
9		algorithm to the app? I'm confused because it says, heart rate …
10	*((points on the yellow piece of paper))*	
11	Aditi:	I'm not sure I'm understanding your question.
12	Joyce:	Basically, there's 3 steps, right? There's something you're
13		wearing on your band, it gets processed, it goes to the cloud...we
14		want to know exactly what we're getting from that algorithm. So,

15 what's the point of the algorithm?
16 Aditi: So what I was explaining to Ning? is that the algorithm isn't as
17 simple as 'within' or 'outside' the range. Let's say, at the
18 minimum, you're asking the band to report data every 5 min. At
19 any one of those time frames, the band recognizes that you're
20 outside the range, ***.15-.2.*** it'll start to report. Every minute...
[Announcement: Myo, follow Khalil.]
21 So after 15-20 min, you got 1 min data production, based on the
22 accelerator, you decide whether this person's heart is beating
23 because he's exercising so based on an activity, or is it that fast
24 while they're sleeping, sitting down or something like that, so in
25 this case it's dangerous. And that gets recorded and translated.
26 Kamilla: ( )
27 Ankita: It's a yes or no answer.
28 Kamila: What's the algorithmic output?
29 Aditi: So it's risk stratifying. Finding people who are outside that range

In line 5 of Example 1, Kamilla reportedly asked Aditi what *i* was. Joyce intervened in line 12 to assist Aditi in stipulating what the three steps should be to get information from the band ("there's three steps, right?). Joyce started to translate the three paths: "it gets processed" [on your band]; "it goes to the cloud" and tried to get Aditi to confirm that "what we're getting from that algorithm" was correct (line 14). But there, Joyce stopped herself short and summarized the problem they were having understanding the use of an algorithm (line 15, "so what's the point of the algorithm?"). Even after re-phrasing it, Aditi did not seem to understand Kamilla's question, so the physician fell back on the instructions she gave Ning (line 16 "What I was explaining to Ning?"). The three steps to which Aditi then alluded did not divide data in the tripartite structure that Kamilla and Joyce needed to carry on their work. Instead, Aditi imagined the data as one bit of information, which the doctor would be charged with interpreting herself: 1) "outside the range" of .15-.2, the band starts recording (lines 17-20); 2); every five minutes, a physician should be able to differentiate between a person's heart "beating fast because he's exercising... or sleeping, sitting down or something like that" (lines 22-24); and 3) results after

recording (line 25). These steps did not translate into commensurate paths that the girls could follow. By the end of the transcript, Kamilla had not gotten her answer, and Aditi was left confused. This situation was partially resolved when Kamilla decided to proceed by making up that information.

Technical *mis*-translations – when the utterances exchanged between speakers cannot make themselves intelligible, such as the kind of discrepancy we see between the "doctoring" of Sirona Care's team leader and her developers' programming schemas – were a direct response to the technology. The Microsoft Band was a wearable device with built-in sensors to detect basic vitals, but it would need to communicate with two other devices to work: the phone and the cloud server. The Sirona Care team had strategically divided their work so that each pair of developers specialized in one of three receiving ends: from the band to the phone app; from the app to the cloud; and from the cloud to the physician portal, for a total of eight communication tasks (Figure 16). By relaying the instructions she gave to Ning, Aditi was not accounting for changes of state as the information circulated on this tripartite infrastructure the developers sketched in Figure 16. The app, which non-coders often thought was everything the hackathon was producing, really came last. By "what's the point of the algorithm," Joyce was hinting that Ning, the programmer writing the algorithm, was only in charge of one step: "detecting *x*," and was not actually building the app to post the encouragements. The source of confusion was visible from the chart: the arrows next to "Phone" had wrongfully suggested that the device itself would be detecting *x* (irregularity) *and* giving encouragement; missing was a path clarifying to Kamilla and Joyce that the algorithm was calculating differences in heart rhythm. To remediate this confusion, Aditi started to sketch the data visualizations for Kamilla on a separate notepad. But '*i*' was still a big unknown.

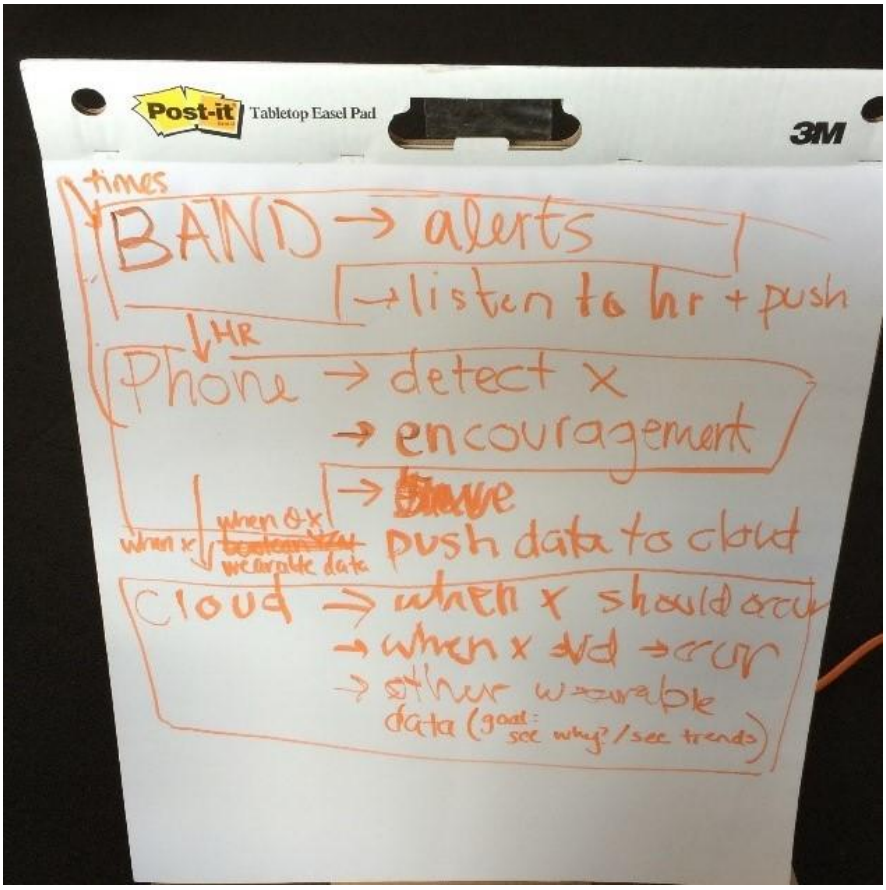

Figure 16 – Sirona Care's Workflow
Source: Screenshot on Apri 24, 2015.

Despite not being able to get Aditi to specify what *i* was, Kamilla cleverly implemented the hack with default parameters while waiting to get her hands on real data. Example 1 gives a hint as to how she could achieve this feat, by thinking in terms of relationships between the four parts specified in Figure 16. As long as their states were interrelated, changing their values would not matter.

The didactic style of Sirona Care's work environment is worth noting. The close approximation to capitalist modes of production at the Grand Hack had not affected Sirona Care's play labor, who was too busy trying to figure the best way to get work done to care whether their hack would find an investor. Instead, the team of nine redirected its attention away from the prize to the physician leader – the representative user for this hack, who was more

likely to bring it back to her hospital. As of February 2017, one university hospital in Kent, England is evaluating the use of the Microsoft Band to monitor people with epilepsy (Microsoft News Centre UK 2017). This suggests that two or three years are needed to witness the commercial adoption of any technology that was play-tested at a hackathon.

More surprising was the division of labor at healthcare hackathons, which relied heavily on the expert testimonies of physicians and let them lead teams with full authority over the end product. This rigid division of labor nevertheless masked the highly negotiated nature of the team's technical workflow. Having complete autonomy over the feature they were assigned, the developers strategically played with the pace of their collaboration. Kamilla hung out most of Day 2, having given up coding the portal because she had to wait for Katrine to connect to the wearable band. Likewise, Ning stopped working once his algorithm was written. Their expertise would not be needed until it was time to integrate the four parts of the hack, and they prepared for it by saving their energy till then. Katrine eventually failed to do her part, losing sleep on it. Luckily, the rest of the team was rested and collectively decided to reorganize their work. As the team was running out time, it was agreed that it would change direction: the team built around Kamilla's mock visualizations instead, writing a catchy slogan, and adding an emulator, which Priya offered to build with Katrine's help. The ability to pace oneself in a collaborative coding setting is thus highly indicative of a kind of disciplined play (see Chapter 5).

I now jump to another team called Reciprocall from the Athena Health hackathon, which in contrast to Sirona Care could not manage its play labor.

Athena Health MDP November 14-16, 2014

The goal of the MDP hackathon was to bring together like-minded, forward thinkers to brainstorm and start developing compelling solutions in healthcare. Athena Health (AH) happened to be a leader in the healthcare IT space and garnered a lot of buzz as the only company hackathon in the Boston area to be in early partnership with MIT. With the launching of the Athena Health's marketplace, "seamless[16]" solutions could now be offered to small clinics (60% of AH's revenue) and hospitals (30%) in administrative areas that included but were not limited to: billing, care management, engagement and efficiency. At the time of my research, Athena Health had just launched an accelerator program to leverage its growing ecosystem of providers (Lindsey, organizer interview, November 15, 2014). Competing with Salesforce, the tech giant that was known for its $1 million-dollar prized hackathon, Athena Health opened up sections of its marketplace and accelerator at its first hackathon on November 14, 2016 to get ideas on possible B2B solutions, and sought the help of outsiders to induce network effects.

[16] Able to be built once, and run from anywhere.

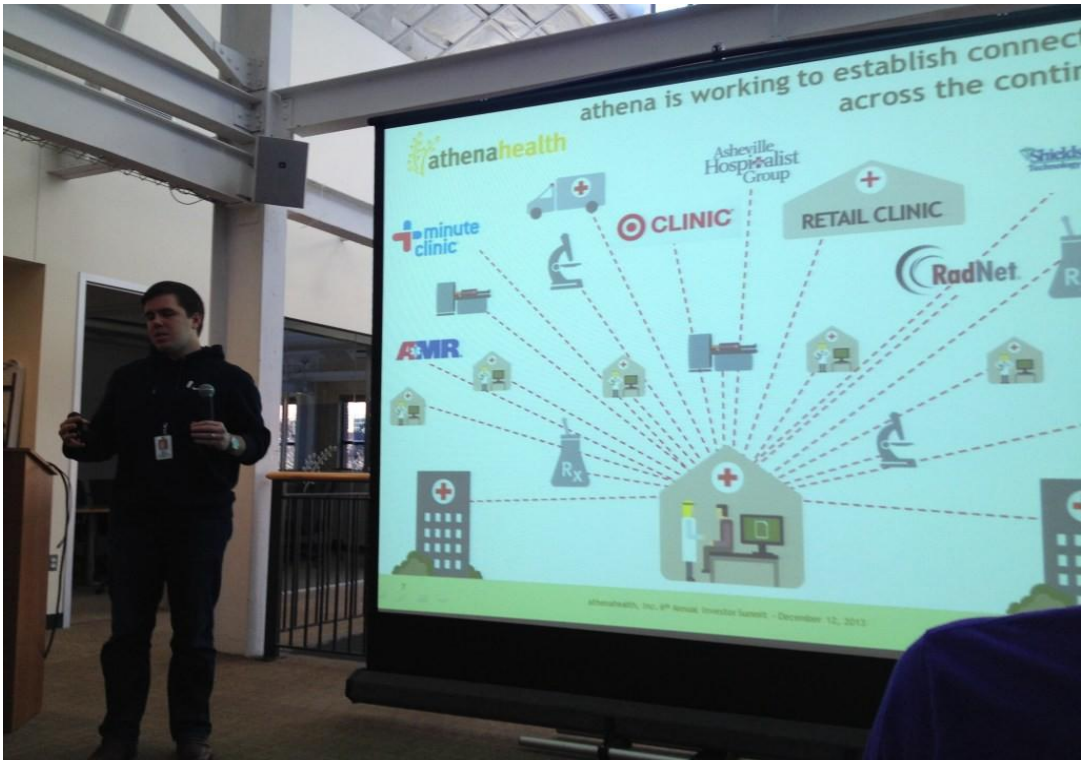

Figure 17 – Athena Health Network Diagram of Its Client-Providers
Source: Athena Health News Website 2014.

There were two downsides to the AH hackathon fervor. For one, the Athena Health API failed to provide any real data or instructions on how to integrate the technology of choice into AH's workflow, so most hacks could not even meet the basic criteria for impact. Second, some could not commute to the campus, resulting in a lower turnout. Table 13 only lists nine hacks in total.

Table 13 – List of All the Hacks Demoed at Athena Health MPD

Team Name	***Summary of the pitch***	***Categorical Innovation***
Carepoint	Multimedia patient engagement survey integrated into EMRs	Patient Engagement
AirCare	Drone delivery of emergency medical supplies	Operations

The Cuff Guys	The blueprint for a new medical device that repurposes component of ultrasound technology into a cuff that can be worn around the leg to speed the detection of deep- vein thrombosis (Third-place winner, received $2000, and office space in the incubator)	Physician Alerts
Engage	Automated text conversations between provider and patient throughout the entire care cycle	Patient Engagement
Kos	Platform to share data with practitioners	Data Exchange
MEDI	A medication reconciliation solution that takes verification work away from the provider, connecting with athenaNet to bi-directionally exchange medication data (First-place winner, received $6000 and office space in the incubator)	Medication Adherence; Data Exchange
Mercury	Pre-visit EMR for hearing loss	Data Exchange
Reciprocall	Networking for diabetics. Automatically calls PT.	Peer support; Patient Engagement

Source: Field recordings on November 16, 2014.

Given the benefit of a market connection, most hacks subsequently lacked creativity. Contrary to the broad representation of apps at the MIT Grand Hack, seven out of nine demos were apps that catered to the needs of Athena Health. Athena Health employees served as mentors at the event, and let participants know when the company might be interested in a hack. The combination of advisors available that particular weekend, and the reward system in place at Athnea Health MDP, may have played a role in the choice of categorical innovation, with Athena Health rewarding hacks that could utilize its network of resource providers. The company also expressed its desire to improve its IT services by regaling winners of its hackathon with mentorship, legal advice, office space and referrals to partners. Guy, from the first-prize winning team MEDI and the Director of Innovation at Phillips Healthcare, had been waiting a whole year for Athena Health to release an API. From such a personal narrative, it was easy to

predict that he would choose to hack around it.

One participant, Tyler Turwick, explained that hackathons were about "soft power." They teach others to think about a problem creatively, and to see how one's code affects other people's code (interview, November 15, 2016). While the last scenario was true, and the rise of Athena Health's marketplace had compelled the company to test the interoperability[17] of some of its services, the reasons for attending Athena Health MDP were different for each individual. An engineer (Andrew, pers. comm. November 16, 2014) cited office space as the best package to win at the hackathon. Another AHMDP participant valued access to Athena Health's network of providers, explaining that AH could help him gain a critical mass of users (Guy Schetcher, interview, November 16, 2014; Vijay Selvaraj, interview, November 15, 2016). Although its organizers marked the event broadly, Athena Health MDP was part communal hackathon, part startup weekend. As a participant and a sponsor, AH scaled back on its own play labor to give participants some autonomy over their future businesses, which it believed was the only way to grow. The accelerator program encouraged winning teams to consider incubating their hacks. Very few could decline to join AH's accelerator. The implications for their hacks were immense in light of these practicalities. The higher stakes at Athena Health More Disruption Please! hackathon structured the play labor of several hacks, and most of the teams tried to plug into its new, albeit very buggy API at some point during the hackathon.

The following case study, revolving around the Reciprocall hack, paints a more serious picture into the hacking process at Athena Health MPD. Given its high stakes, the Reciprocall team had recourse to an altogether different type of play labor as its physician leader and Athena Health taught the diverse members of the team about current practices and possibilities in digital

[17] The ability of computer systems or software to exchange and make use of information.

health.

Role reversal: The case of Reciprocall

The Reciprocall team wanted to design a peer-based tiered support system for diabetics. The telehealth model was different from Sirona Care's. The hack entailed writing a program that would match patients by age, symptom, treatment program, and socioeconomic status, and schedule them to call each other to report on difficulties and improvements on specific health outcomes. It was the reverse of what the Sirona Care had placed within the doctor's purview.

More specifically, Reciprocall worked on de-colonializing the authority of the doctor in population health programming. Its team, consisting of economists, population health experts, a nurse, and two developers, totaled nine people. The doctor on the team, Jessica, ironically led the brainstorms, even if it was her first time at a hackathon. This suggested that the climate for healthcare innovation, namely its legality, could *in production* deter developer input on the basis of their ignorance of the finer technicalities involved in treating a disease and safeguarding patient confidentiality. After all, the premise of healthcare hackathons involved real profit-making activities, directly tied to ideological modes of capitalist production. Whereas technical expertise was welcome in certain areas of social life, no one would trust a developer's judgment inside the walls of a doctor's office. Despite these constraints, The Cuff Guys were exceptionally more agile and more disciplined in their play on account of the relative absence of a doctor on the team. Conversely, Reciprocall remained disorganized due to it team's emphasis on data. When the Reciprocall developers got stuck, Jessica awkwardly responded by asking everyone to give feedback in places where a developer's leadership would ironically have been useful. Against Jessica's mission of inclusion, I would argue that hackathon teams never progressed when they strove for consensus every step of the way. Jessica was very methodical in the

afternoon of Day 1, but could not adapt the distribution of expertise to any particular problem.

Example 2 shows the degree to which the team had to learn from her about current healthcare practices. The heterogeneity of the team and the struggle of its members to grasp the value of the hack, led to many teaching moments throughout the hackathon. In the case of Reciprocall, a high learning curve was detrimental to its productivity once Gerard left. Gerard could afford to teach them because he had experience facilitating groups at Athena Health and excelled at managing tangents (line 15, "I think this is kind of a sidetrack brainstorm."). But the brainstorm in Example 2 jumped from topic to topic; the transcript shows no alignment of goals the longer they talked. Paul started by asking the question on everyone's tongue: "If I was a doctor, why would I want this?" (line 1) Gerard proposed that the value be to governments, in providing metrics (line 5) that met their criterion for "meaningful use" (line 4). The project manager enlisted the help of Simon in line 8, who very much like Aditi, complicated the picture by stating that there were "different stages and sets" of criteria that were written to "implement more electronic health records in the country" (line 9). Meaningful use had something to do with the organization of electronic health records and its political consequences. Paul, a financial analyst for a bank, immediately raised questions about tax rates and incentives to providers (lines 10 and 16-17). He was later shown to possess extensive knowledge of the availability and outcomes of corporate wellness programs in the US. To Gerard, however, the doctor had social trust (line 19), implying that she would most likely be a customer.

Example 2

1	Paul:	If I'm a doctor, why would I want this?
2	Gerard:	At the higher level, it's… so when governments are mandated that there is
3		an uptake all these technological services for the healthcare space, in order
4		to actually get quote-on-quote "meaningful use" out of the system, they
5		created certain metrics – I'm interpreting wrong, there are certain metrics
6		that will show "okay, you guys have the technology, but you're actually

7		hitting these meaningful… numbers" and there's different sets…
8	Simon:	Yes. There are different stages and sets, but it's a plan to implement more
9		electronic health records into the practice throughout the country.
10	Paul:	Is there a tax rate?
11	Simon:	So it's direct cash, and eventually there is a penalty with Medicare
12		reimbursement. So there is a cash incentive. Until 2016 (.)
13	Jessica:	This whole model, the hardest thing is to get the patient (.) to actually
14		Meet them where they are and engage them.
15	Simon:	//Maybe this is kind of a sidetrack brainstorm…
16	Paul:	//What I think is neat about this, the providers have the incentive to use
17		this. For the doctor, it's a mecca of information, the nexus to the right
18		answer.
19	Gerard:	Right, the doctor has social trust.
20	Jessica:	((Talks about a framework, a safety net)) The doctor can't do for the
21		patient what the social support can do for this patient. The healthcare
22		system can never afford to do that. What do other people think?

I included this transcript because it lays out all the issues that some badly assembled teams faced at hackathons. Not everyone could play. But Jessica's initial pitching of a telehealth-based program for diabetes made her a visionary, in touch with what was coming in the hospital and small clinic communities. One panel at the 2016 Seattle Startup Week, which I attended a year later, considered leveraging the city's ecosystem of partners in healthcare innovation to strengthen the cohesiveness of behavioral-based outcomes (panel discussion, October 2, 2016). But Jessica had already predicted this, suggesting that play labor was inseparable from the mode of capitalist production already in place.

Example 3 featured Jessica challenging Simon's assumption that doctors were the primary actor in population health. She asked how his scenario would work by inquiring into the imagined persona whose job he thought should be of classifying people (line 3 "how are you going to put people in different categories?"). Inadvertently, Jessica had instigated her own brand of play labor by suggesting a role reversal (line 2, "do you think you want a doctor in there?"). If the doctor was out of the equation, who else in the ecosystem of healthcare existed to manage

chronic conditions? Simon tentatively offered that triaging should be a receptionist's job (line 4, "Sounds like a receptionist job."), but Jessica was far from satisfied with his answer. She drilled him with questions on how he "actually" envisioned two patients in communication before she concluded that a user experience expert might be better qualified to draw this relationship for them (line 9). Her virtuous, militant conduct was visible from this passage:

Example 3

1	Simon:	But I think a doctor should be able to navigate that.
2	Jessica:	No, I think… do you think that you want a doctor in there? How are you
3		going to put people in different categories?
4	Simon:	Sounds like a receptionist's job. Like when they're asking, do you
5		smoke?
6	Jessica:	You said doctor, I thought.
7	Simon:	My bad. I meant provider.
8	Jessica:	How people will pay to use it? You can ***actually*** envision two patients
9		communicating? That's the user experience expert. ( )

Looking for qualified experts was a common practice among doctors, who must be accredited to examine any part of the body (Minh Dau, pers. comm., September 3, 17). Jessica had brought her professional ethic to the hackathon where she was expected to transcend it. While doctors cannot be expected to do the work of an office manager, her leadership style of looking for credentials made even less sense at a hackathon.

The research phase engrossed the team. The hack do's and don'ts in Chapter 1 warned against this ("Do not get stuck in the ideation phase"). The collaboration was not the best use of the group's intellectual resources, but it did speak for Jessica's ability to be engaging as a leader. If the cadence of the team's conversations contributed any insight to play and work at healthcare hackathons, it is that the reversal of roles opened itself up to a myriad of possibilities. The team enjoyed learning about the people and policies that supported doctors and never ran out of energy like other teams did.

Pivoting: The Cuff Guys

The Cuff Guys from the AH hackathon raised new questions concerning the kinds of play labor involved in cleverly reinventing a technology to meet social requirements. A process known in the startup world as "pivoting," or simply changing ideas based on feedback, pivoting was a strategy that required consensus from all team members upfront, which in the case of The Cuff Guys team of three was relatively easy to get. Pivoting also demanded much more discipline to optimize a team's intellectual resources. We see in the case of The Cuff Guys that complementary expertise in allied engineering fields came in handy in facilitating the type of disciplined play in which they engaged.

The Cuff Guys was made up of three biomedical engineers and one developer interested in applying his programming skills to several industries. Andrew, who had visited all the way from Yale, was initially worried that their first hack might be too "code- intensive." Their final idea, which won them the runner-up prize, was a cuff that detected blood clots.

Code-intensive hacking sessions could sometimes lead teams into a dead alley. Reciprocall had relied too much on its ability to extract patient data from a company API and its developers fumbled along with the rest of the team through the ideation phase. The latter did not pivot in part because they were not disciplined enough to come up with another idea, but also because they could not find any use-value in a hack that did not use real patient data to test its assumptions. After many attempts to extract that data from the Athena Healht API, Ying dropped out of The Cuff Guys. Andrew must have sighed out of relief; The Cuff Guys took advantage of their opportunity to stop coding then. Before leaving for the night, they had pivoted with a cuff that better matched their expertise in hardware.

The Cuff Guys' initial setback was common among digital workers in the field of

healthcare, where HIPPA laws limited the free use of patient data. After receiving their award, Khalil remarked that his team had spent the least amount of time on their hack out of all the other teams at Athena Health MDP (pers. comm. November 16, 2014). This statement was further testimony that smaller teams had an advantage over larger ones; they could more easily reorganize their work whenever they get stuck.

Chapter 3 Conclusion

This chapter looked at three teams – Sirona Care, The Cuff Guys and Reciprocall – from two hackathons in healthcare. The types of play labor covered were mistranslation, role reversal and pivoting.

The Sirona Care example illustrated the unruly nature of collaborative coding. Play labor for Sirona Care was directly related to the risk and uncertainty built into its workflow. Kamilla and Aditi's fragmented knowledge was apparent by their confusion over *i,* a convention unfamiliar to Aditi, who ended up mistranslating it. How the hack was magically coded without this knowledge would give any non-coder pause. But a programmer like Kamila, who understood that she could mock up values first and substitute them later, did not stop there and improvised. This coding practice is common to data visualizations, which had not yet been implemented in physician portals but was one simple diagnostic feature of wearables that a team could offer, which eventually saved Sirona Care.

A preference for hardware at healthcare hackathons begot larger teams, corresponding to most innovations in the healthcare industry coming from medical devices (PwC 2015). Sirona Care creatively made up for its size. The team divided their labor such that two worked on four standalone features at a given time. Sirona Care might have exhibited a rigid division of labor,

but it sandboxed production so that one pair of developers would not risk breaking other teammates' code. Aditi's sensible delegation style rewarded the Sirona Care during demos, but only because, again, members on the team were autodidacts and knew how to pace themselves. I will return to the team's disciplined play methods in Chapter 5. In this chapter, Example 1 indicated that Kamilla knew how to work around the constraint of not knowing '*i*'. Ning also knew to stop coding after he wrote the algorithm, reflecting both developers' uncanny ability to coordinate other people's labors from their own vantage points. This would confirm Lazzarato's (1996, 136) observation that entrepreneurial subjects are now required to manage the structuring of the social cooperation of which they are a part. Discipline in collaborative coding is immaterial labor performed by Sirona Care that differentiated its play from Reciprocall's, which could not manage its size, and required too much research at the Athena Health MDP hackathon event itself.

The more successful team, whom I thought for a minute would quit, was The Cuff Guys. The three biomedical engineers were as strategic as Sirona Care, but they got rid of coding altogether. Their play labor, conceptualizing a hack that matched the team's expertise in hardware, went on to win recognition as one of the most promising hacks to come out of Athena Health's first – but reputably disorganized – hackathon. The Cuff Guys were an exception because they were strictly engineers; not everyone could pivot as they did the night of the hackathon. Their example suggests that it took a certain level of discipline – and consensus – to achieve this feat. Reciprocall from the same hackathon could not play. But its physician-leader made a very interesting proposition in Example 3: instead of removing the coding, why not remove the doctor out of the program? By flipping the logic of population health programming, Jessica suggested a clever hack based on the reversal of roles – a singular type of virtuous play

through language. Together, the three team's play strategies constitute different forms of play labor because their labor responded to the specific material constraints of their hacks at the same time as it focused on the demand for capitalist production in digital medicine. Two received funds to launch startups, and the third predicted a trend in telehealth for the next year.

I risk contradicting myself by stating that the different responses of teams in the absence of specific information on data types produced different strategies for getting things and in no way determined that one's code limited to one kind of play. As they ran more risk in this industry, teams assigned authority to the physician leaders, who enforced autonomy and provided a rigid structure for work. But by providing a structure, they manifested the very relations that their teammates could flip. This convention was not true of journalism teams, who adopted a more developer-centric style of leadership, which I discuss in the next chapter.

Chapter 4 – Dissensus in Journalism

The non-linearity of video news has emerged as a real concern for journalists in recent years. In a blog piece on new models for reporting, Fisher (2017) offers a meta-commentary on how the Internet has undoubtedly sped up the news cycle, and the incremental story— as a result of daily updates that fragment the news — has so far failed to keep up with its pace.

> "Let's articulate the problem. There are too many damn Trump stories. … [N]ewsrooms self-organize around the 800-word incremental story: as events develop, newsrooms write new incremental stories at each significant moment. As these collections of incremental stories grow, you end up with a disorganized group of stories that represent snapshots in time. It makes following a story that develops over time difficult to understand audiences are clamoring for a more distilled, high-level view of everything happening at once. Other newsrooms have tried to organize around an alternative story model before. Vox's Card Stacks are a promising, if under-utilized attempt to track developing stories. Axios's stream-like homepage tries to make the river of stories more scannable. Both of these attempts came at the start of a new organization."
>
> *Source:* Open News, blog, March 31, 2017.

The *New York Times* cited the quest to make news "freely available" (without a paywall) as the main driver for incremental stories. *The Times* wanted to make itself "a little different" from the its competition (The New York Times 2017) and was turning its attention away from the "dutiful" mission-driven pieces for which it had been known because similar versions were available free of charge elsewhere. To match and anticipate the habits, needs and desires of their readers, present and future, *The Times* proposed that: 1) reports be more visual – it recommended its staff to become more comfortable with photographers, videographers and graphics editors playing the primary role in covering some of their stories; 2) more innovation be fostered like they had in 2014 with the development of bite-sized news, which cultivated a loyal subscriber base; and 3) journalists shall use a more conversational writing style, hence the news page 'What the Fuck Happened Today?'. But it also meant that, 4) they would accelerate hiring the top journalists and people with skills in multimedia digital production. Two years prior an internal

report had been leaked and frequently cited by my journalist-informants that year, establishing that its competitors were newcomers Buzzfeed and Huntington Post, barely ten years older than 1854 *New York Times*. As the leaked report had implied, Buzzfeed generated content from the virality of memes and had a far more effective business model for digital journalism, premised on the mastery of new media tools. But in its two-year retrospective (unleaked), *The Times* had announced that they had grown weary of haphazard training – a key insight since most of that training had happened at hackathons.

The spectacle was directly incorporated into news evidence. On the one hand, ideating and creating is no different from what Lotte Hoek (2014) has observed Bangladeshi film buffs do: an internal reconstruction of one's sensorium that altered the ways people participated in the world. On the other hand, these tech-savvy journalists were learning new ways to mobilize the senses, not by immersing themselves in new spectacles of evidence but by participating and building new capacities for new media; not to make into their own but to appeal to this digital reader persona's aesthetic. As Fisher (2017) has shown, spectacles of evidence can be as simple as ingesting a caption feed to live video and dumping the comments into a Google doc for a group of reporters to annotate and render into a fuller experience, producing a lasting resource that a newsroom can produce and have readers choose it over what is freely available. The case of Newstritious below is about one team's desire to advocate for the power of a reporter's training and editorial judgment, different from Buzzfeed-like incumbents by the former's central ability to discriminate between junk news and quality reporting based on superior journalistic standards.

The Hacking Journalism (HJ) series that make up the subject of this chapter fell in the category of a communal hackathon. These events had for primary purpose to develop resources,

infrastructure, practices, and culture for the community of data journalists facing the challenge of advancing methodological approaches for newsrooms. The organizers of Hacking Journalism Boston had attracted participants by advertising it as a means to explore specialized methods under "future trends in mobile journalism." The sequel, based in New York City, was advertised internally as a professional development opportunity. The participants qualified for this opportunity because they were members of an established community of data journalists around the world called Hacks/Hackers. Spearheading the Hacking Journalism hackathon series was Kawan Virdee, a developer at Embed.ly, who had been inspired by Media Hack Day in Berlin in the months preluding HJ. Kawan wanted to bring something similar to Boston, and teamed up with Matt Carroll, then a researcher affiliated with the MIT Media Lab's Center for Civic Media, to host the first journalism hackathon series. The first one took place in Boston on June 7-8, 2014, with world-class sponsors, Enigma, Twitter, Parse.ly, Embed.ly and Bloomberg News.

It should be noted that three out of five sponsors were in the business of data analytics and not directly involved in news production. A startup sponsor, Enigma, illustrates another way that hackathons can be used: by entrepreneurs to improve their branding. The data startup, which operates as a warehouse for over 100,000 datasets from governments, organizations and businesses, received $4.5 million in series B funding from *The New York Times*, Comcast, and American Express two weeks after the hackathon, and was admitted into the FinTech Innovation Lab in New York City that same month. Their participation at a hackathon, which counted at least three representatives from the company, was in direct relation to their activities within the startup ecosystem of the mid-Atlantic region. The hackathon helped put the startup on the map and provided a means to test their Application User Interface (API) with a target demographic, namely the journalists. Bloomberg News, an established newsroom alongside local *Boston*

Globe, had only one representative at the event. Compared to startups, traditional newsrooms had not yet caught on to the value of this new hackathon phenomenon – a fact the organizers remediated six months later at their second hackathon in New York City, with participation from nearly all newsrooms in the city.

Two startups illustrate what the startup ecosystem had to offer hackathon participants. Embed.ly had been in the business of engaging with journalists with Card Stacks. FOLD had a similar concept and was developed by MIT students in engineering and the humanities. The content had most likely been vetted by their MIT Media Lab advisor Matt Carroll himself, who gave a Lightning Talk at the beginning of Hacking Journalism Boston showcasing his favorite project. Embed.ly's data analytics dashboard was the most popular at the hackathon and saw some of the most practical applications by teams: four teams used the Embed.ly API compared to five for the Twitter API (see Table 14 for a list of the hacks with API integrations).

Card stacks expressed a demand for more context in the news (Peterson 2014). Its promise as a solution to disruptive incremental storytelling received much attention in 2014, when Embed.ly competitors Vox and FOLD emerged on the scene. It was, however, the subject of reporters' disappointment from their underutilization in 2017 (Tyler Fisher, March 31, 2017). Figure 18 is an example from FOLD containing two cards from the stack. On the FOLD website, I clicked on a headline that took me to a page on a separate tab, tabulated with Wikipedia-level information on the decline of the city of Jakarta. There were a total of ten flashcards with links to videos and other articles. Figure 19 shows a slide from the Vox card stack on Obamacare, which an employee claimed on the website had been written and updated by professional Obamacare reporters. Vox reporters had license to use a conversational style of reporting, and did so by playfully building up the reader to look forward to the most critical piece of information on card

no. 5: it surprised the reader with the most irrelevant definition of the slang term “man date” as a play on the word “mandate” in Obamacare’s provision called “the individual mandate.” Play labor in journalism was directly embedded in the content.

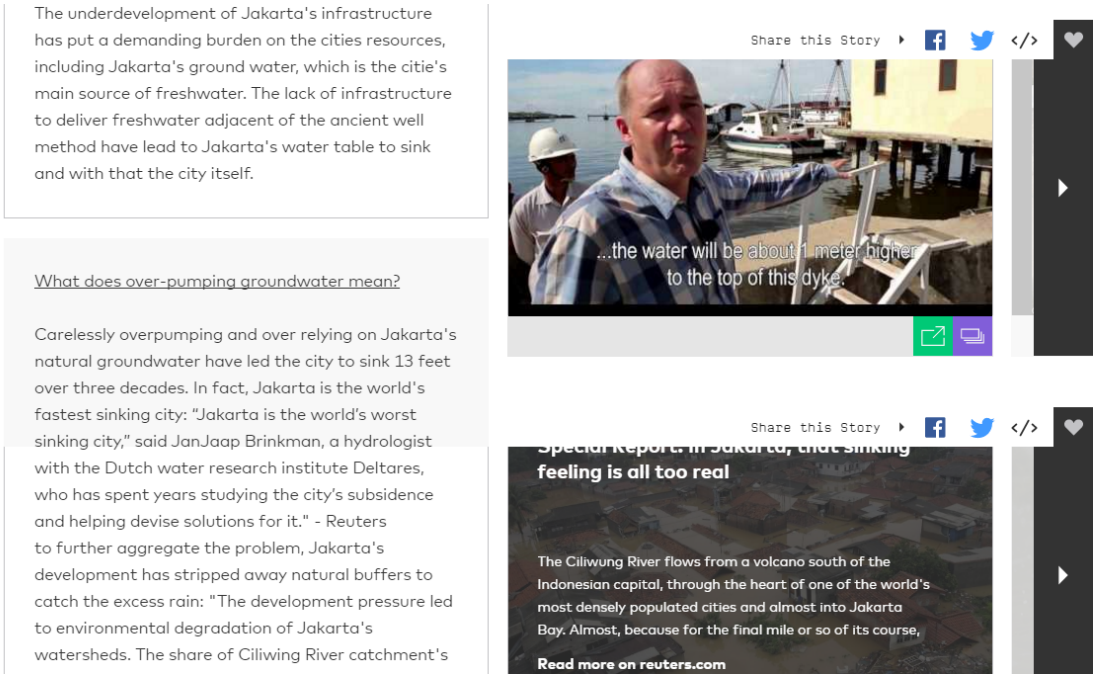

Figure 18 – FOLD Card Stack “A Mega City on the Verge of Collapse”
Source: Tinoco 2016.

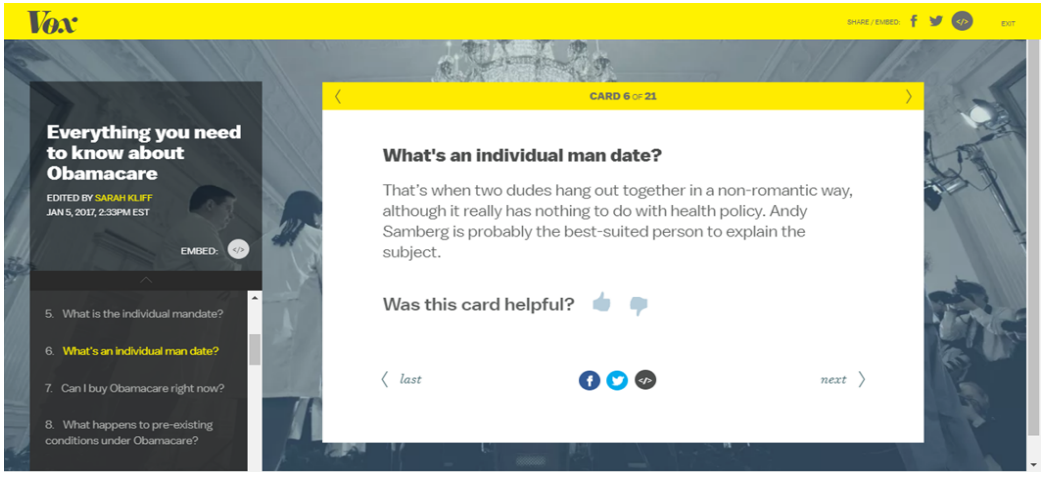

Figure 19 – VOX Card Stack for Obamacare
Source: Kliff 2017.

The 100-plus journalists who came at the first Hacking Journalism represented Bloomberg, Conde Nast, Vox, *The Boston Globe*, and *The Guardian*, the latter of which had flown journalists and developers from London to get more acquainted with staff from its newly opened NYC location at the hackathon. Hacking Journalism was designed to attract industry professionals and get them to address newly voiced issues in print-centric journalistic practices. Some participants had already been exposed to the hackathon form and worked on creative hacks. For instance, *The Boston Globe* team that developed the News Bingo app had participated in its own internal hackathon during the elections.

The subjects of the lightning talks, which opened the hackathon, centered on several trends in digital journalism that the speakers – mostly journalists - believed would impact their work soon. Two of the lightning talks addressed the gaps and possibilities in sensor journalism for tracking air pollution and water quality. Matt Carroll tried to answer: "how do you tell a complex story that requires a context?" followed by Jihii Jolly from the Future of Journalism Project, who talked at length about the lack of human-centered design in algorithmic news. Nicely complementing Ms. Jolly's talk, a certain Frankie described a digital newsreader's reading habits, and the different formats in which she might receive her news throughout the day. Frankie concluded her talk with an alarming conclusion for the journalists in the room: customers were moving away from journal subscriptions, adding extra pressure on newsrooms to retain their subscribers. Finally, Matt Carroll rounded off the Lightning Talks by offering five new trends in digital journalism: collaborations in the newsroom; participatory interfaces; bite-sized news; changed mode of delivery; the distribution to the Google Glass; and mobile news platforms.

Table 14 – List of All the Hacks Demoed at Hacking Journalism Boston

Team Name	Description of the Hack (By the Team)	API Used
Atom	A surprising daily news article from your Twitter feed	Twitter
Consolidated News	Get all similar news in one place	N/A
BombPopper	Rate your mood while consuming news to keep track of how news consumption affects your mood.	N/A
AiDatacle	Personalize and make news stories more relevant. It can drop a tag to fill in a sentence about how your local rep voted in a story about legislation.	Twitter, Enigma
Source.Me	Source.Me scours Twitter to uncover the top authorities on your search topic	Twitter
Amgine	Using the Enigma API, a mobile solution to visualize public data in user-friendly graphs.	Enigma
News Bingo	People want to share their photos and have hashtag conversations with their friends and followers. News organizations are being left out. Twitter Bingo to encourage users to submit photos that fit in a category.	Twitter
Accio!	Create a tool for making callouts that include a light interactive element as a gateway to asking for more detailed information.	The Guardian API
Triangle	Natural language processing of news articles while users are reading	N/A
Collater	An organizational tool to make journalists' lives easier.	N/A
Parse.ly Analytics Hub	A dynamic web analytics hub using Parse.ly API and D3	Parse.ly
Newstritious	A Chrome Extension that uses your browser history to generate visualizations of what news you consume. Inspired by the proliferation of personal accountability apps in the world of food and financial health (like FitBit, Weight Watchers, RunKeeper and Mint)	Embed.ly

InLine	InLine is a news-sharing service with inline comments. Users can submit URLs from around the web, annotate articles other users have shared, and recommend related articles. Think Rap Genius meets Reddit.	Embed.ly
Main Street Journal	A hack to liberate paywall news articles from the Wall Street Journal (aka "mugging Murdoch every day")	N/A
They're Watching Me	Watch the watchers	Embed.ly, Twitter
HiNeighbor	Mapped news by geolocation	Embed.ly, Boston Globe API, Instagram API

Source: DevPost

Table 14 lists all the hacks demoed at Hacking Journalism Boston. As can be gleaned from their summary descriptions, the teams were preoccupied with making the news more relevant to the readership, and developed hacks to make it easy for journalists to personalize the news; report those affected by location (Datacle and hiNeighbor); visualize the consumption of the reader, and encourage him to read more widely (BombPopper and Newstritious) or her to share (Inline and News Bingo). The diversity of hacks accounts for the relative ease with which journalists came up with creative applications for metadata that was now freely available to scrape on social media. In the process, they established new rules for writing and circulating the news.

Despite the theme of the hackathon being centered on smartphone apps, most hacks were not inspired by mobile trends. A potential factor in this decision to disregard the theme was suggested early in the brainstorm of Newstritious: news organizations might not want to be limited to mobile apps. Instead, the team pushed for responsiveness – good practice coding an app with adjusted dimensions for all devices (mobile, desktop web browsers, and iPads). Also, a number of Google Chrome plug-ins proliferated in that summer period, with co-organizer Kawan

and Newstritious learning to code their own.

Newstritious

Figure 20 – Newstritious Dashboard with Mock Data Visualizations
Source: HackDash

Newstritious was a Chrome Extension that produced visualizations of one's browser history to track their news consumption. On Hackdash, the team recorded a pitch describing the hack as "inspired by the wealth of accountability apps in the world of food and financial health." The dashboard was captured with mock data for a single user (Figure 20). Despite not ranking among the top three hacks, Newstritious had the most compelling pitch and was frequently cited in news publications. Matt Carroll, who had tracked the projects as they evolved throughout the event, summarized the team's contribution on June 7, 2014 as "a Google Chrome extension that generates visualization of the types of 'serious' and 'junk' news people read." *The Guardian* described Newstritious – a portmanteau combining the words "news" and "nutritious" as a playful take on the fitness tracking device movement, with sales of Fitbit and Misfit booming

that same year.

> **Newstrition (“You are what you read”)**
> Health and fitness apps are a big deal at the moment, enough so that Apple have added an integrated Health app to iOS 8. Could we apply something similar to news? How “healthy” is your reading history? Are you reading too much “light” content and not enough hard news?

Source: Berry and Cevey 2014.

The team was made up of two journalists, Katie and Jihii, who had given a lightning talk earlier. They paired up early during the team formation phase and recruited designer Sacha, actively looking for developers next. Ying hesitated to join their team on the spot, a common habit for developers who throughout the years had learned to wait before they settled on a project. A developer strategically visited as many teams as possible in search of the perfect combination of creativity, impact and feasibility in a hack. Alex, who carried his banjo around, was the first developer to join the three women and the rest of the team assembled itself, with Kyle, then a front-end developer for Embed.ly, coincidentally standing by their table and Ying finally agreeing to join their team. Together they made a balanced mix of expertise, which was found to make it easier to delegate tasks without overwhelming the others.

Developer leadership

Example 1 (courtesy of Beth Semel, whose fieldnotes were invaluable in this research) shows Ying employing a didactic style of brainstorming, using specific techniques to direct the Newstritious team in thinking of a problem they wanted to solve. Developer-centric leadership was a form of play labor astutely different from the other teams’ in my sample because developers contributed a highly sought-after coalitions of skills (Gershon 2007), which included a sophisticated managerial style that emerged from one’s technical domain expertise.

Example 1

> Jihii wants to know how time consuming it would be to use multiple sources. Jihii says that "generating recommendations" is on tier two. Alex says let's write out a list first and then categorize. Jihii describes to Ying this second tier concept (comparative, compare people's news diets). Alex adds some insight that Katie responds affirmatively.
>
> Then Jihii says, ok the third breakdown could be the editors analysis (the editors apply their own filters – junk, healthier) and (Katie says) that would be where suggestions come into. So, Ying says, we need to add in another breakdown. And that part, he says, would be the hardest part. The other tiers (the pie chart) we can get for free from embedly, but the editorial part requires us to figure out what is "Bad," what is "junk." Everyone, he says, is trying to figure out what "junk" is – what is junk, that's like the billion-dollar question. He says, that's where I would probably draw the line. He says the "what is junk" question is something that could be answered in its own separate project – if they want to approach "what is junk," they can't really aim for accomplishing much of anything else. He asks (somewhat rhetorically_ if we have any idea how to, in an automated way, identify "media junk."
>
> "Show me!" Ying says.
>
> Jihii gets out of her chair. She begins to draw out what they were thinking of. Sacha starts.
>
> Ying asks Alex what is background is in. Alex says he does backend, Javascript… Katie is talking with Jihii and Sachia simultaneously. Jihii asks how "you guys" feel about that. S gets up to make her own drawing. Ying and Alex are discussing which skills they individually do and don't have. Kyle realizes what they are talking about and then pulls up his chair (it seems like he can do precisely what Alex and Ying can NOT do).

The passage above was an example of developer-centric management, whereby an individual – Ying – gained authority through his technical translations of his teammates' journalistic values. Before Example 1, Ying had successfully, albeit pedantically, managed Katie's expectation that all was traceable on social media. He cited his colleagues' difficulty at work, trying to track e-mail campaigns to explain to the journalists on the team that her bias was unfounded. The team eventually decided on a Chrome Extension because the hack appealed to the developers' sensibilities. In contrast to the repetitive tasks that data journalists performed on a daily basis (i.e. scraping websites), Alex and Ying were more inclined to take a creative approach to getting work done: they judged the merit of a hack in terms of its feasibility and innovation. Since they

had never built a Chrome Extension before, and it seemed a natural direction in the field of digital journalism, they felt it was the best use of their time at the hackathon. Ying dominated the conversation on those grounds; the energetic 28-year-old got the team to map their thinking and draw the line between what would be an amazing experience to build and the virtually impossible (Figure 3). More importantly, he was able to frame a catchy problem for the journalists: how does one go about designing an application that can automatically categorize the news as "junk"? What constitutes "bad" news? He warned that not having a junk category would be a big problem for developers in the long run, and that they – the journalists – could easily be pigeonholed into answering that "billion-dollar question."

Ying's pragmatism contrasted with Alex's reportedly soothing presence, which balanced the "energies" of the collaboration and effortlessly kept them on track. A lot can be said of the emotional labor performed by Alex, whose style of management was later noted by Jihii as helpful. Alex did not aim for a spectacle like Ying did, but made everyone feel included. He also established the pace for the team. At the beginning of Example 1, Jihii is seen sketching. Seeing her drawings, Alex stopped her and invited her to write out a list first *and then* categorize them. The instructions uncannily mimicked the steps that the developer eventually took. They had to build arrays off of lists to call, and transform the categories into indexes – the id's that the articles were given when they were uploaded, which made it easier for the developers to find or count an article according to its category. In that particular example, Ying translated and negotiated goals while Alex instructed the journalists on the steps to take procedurally.

During the brainstorming session, the Newstritious team members discussed technical constraints such as bypassing personal Gmail accounts to only download browser histories from news websites. Getting deeper into the concept, Sacha and Jihii came up with a diagnostic

hierarchy of journalistic intents. A Twitter evangelist and mentor praised the idea behind Newstritious, finding it a much-needed tool to capture immaterial labor on Twitter. "There is no mechanism for editorial judgment for Twitter."

The journalists exhibited a high level of business-savvy and technical proficiencies. They understood what user flows were about. Sacha, Jihii and Katie were interested in reinventing – or "hacking" – their traditional role as gatekeepers of information. They imagined "marrying" daily logs and recommendations to help users learn their habits from them and balance their news consumption. Later, Katie would report the impossibility of removing the journalist from the equation: on a spectrum of digital labor, the journalist ensured that there was a job for her out there by selling off her editorial judgment, and the reader's attention provides the raw material out of which she produces visualizations, from co-valorized informational goods. Sacha, Jihii and Katie seemed to envision themselves as optional intermediaries in this on-screen dynamic, displacing their jobs elsewhere. By not requiring any active downloads, their role would not be limited to their own platform, but draw from other collaborations with other news and social media platforms to bring in revenue and stay competitive.

In sum, the journalists theorized a principle of behavioral change that reflected their daily frustrations working in an industry that had become increasingly reliant on users to produce the cultural and informational content of the news (Lazzarato 1996). Tweets, shares, likes, and other speedy feedback mechanisms were threatening livelihoods (Boyer 2013) by requiring that the best picked up new skills in digital production and data science. Newstritious was a win-win situation in their lifeworld of algorithms and bots; the digital currency of these journalists was established as their creativity in reworking the relations of production to need them. What's more, Katie and Jihii were reinforcing their affinity with developers. As Jihii pointed out,

journalists these days had taken just enough design and coding courses to be able to tell the developers they wanted. While the no-bullshit leadership of Ying and the soothing presence of Alex kept the women from any possible diversions, Jihii, Sacha and Katie's drawings punctuated a developer's workflow with that indirect, reflexive approach to commanding authority on their part. One notices from observing the collaboration in its entire duration that authority shifted throughout the day of the hackathon; the journalists pitched the idea, at which time the developers listened, but they later were seen conducting research around the developers' need for mock data.

Non-linearity: Video Pizza

I received an invitation to the next journalism hackathon in NYC from Jihii, whom I decided to follow because she was going to implement an idea she had held since the first Hacking Journalism in Boston. Even though the decision fell upon the whole team, her contemplation of an app that took the temperature of tweets to gauge interest in news content had gained favor with her then-assigned group.

The same set of organizers from the first journalism hackathon had wanted to improve the experience of participants by assigning teams to ensure the balanced mix that Newstritious had achieved. This strategy backfired, and the case of the Video Pizza's merger of two disgruntled teams is proof that assigning teams could also negatively affect their dynamic.

Some of the topics discussed at Hacking Journalism NYC revolved namely around the relationships between empathy and interactivity in video storytelling. Storycode founder Aina Abiodun and Ingrid Kopp, Director of Interactive at Tribeca Film Institute, talked about the theory and techniques inside interactive videomaking, while Playmatics game designer Nick Fortugno delved into the mechanics of interactivity in video games (game mechanics, interfaces,

interactions, variable systems, and state machines), which he argued were narrative tools that can communicate stories through play as much as through camerawork and writing. Lam Thuy Vo, one of the organizers and an investigative Journalist at Al-Jazeera, deconstructed the video medium in defense of really, really short videos. Presenting examples of how to "hack" video allowed spectators to imagine new ways to work with digital media, to the possibility of reporting in multiple formats.

Storify had been a crowd favorite at Hacking Journalism NYC and the concept took off when Axio later offered the capacity to scan daily logs. Developed outside a hackathon, the business performances of Storify and Embed.ly's Card Stacks indicated that the hackathon was primarily a place to explore future trends and that the work of implementing them is largely a post-hackathon affair.

Table 15 – List of All the Hacks Demoed at Hacking Journalism NYC

Team Name	**Descriptive Slogan**	**Categorical Innovation**
Full Sight	A simple tool that allows journalists to use video as texture in their work. Highlighted text will subtlety fade in short video clips.	Multimodal Literacy
Ojos Ven	A secure platform for anonymizing and sharing video shot on people's cell phones using Instagram- like filters.	Privacy
Resume	Resume facilitates media consumption and loyalty by making it a seamless experience across all platforms.	Cross-platform
Mystery Science Theater 4K	Soundcloud's time-based comments but for video and on steroids.	Multimodal Literacy
Video Pizza	Sifts through an entire video to find its richest portions and delivers them in easily-consumable, shareable "slices."	Skip

Tmp.RTr.ly.io	Take the temperature of a video by running sentiment analysis on top words in tweets around a shared video. Useful for publishers who need metadata to inform video recirculation. Useful for viewers who like colors and emojis.	Meta-data
Catch Up	Browser-based app that allows the user to watch videos of a single news event as a timeline.	Resume
Cume	Pulls video from live events and allows the audience to watch and discuss events as they unfold. Allows online content producers to curate an experience	Commentary
Greyhound	Accelerated video discovery	Skip
Suncast.io	Trackerless torrent for sharing video.	Anonymity
ShoutBoost	Submit, discover, curate, and promote breaking news videos across social platforms	Curate
Scrubber	Tinder for videos. Use gestures to scrub and scroll through multiple videos.	Skip

Source: HackDash

Table 15 lists all the hacks from Hacking Journalism Video in NYC. Unlike mobile, video was not seen as a platform, but rather as another tool that journalists could wield to accelerate video discovery (Video Pizza, Greyhound); support special needs for live; for multimodal commentary (Tmp.R.tr.ly.io, Full Sight, Mystery Science Theater 4K, Cume); protect anonymity (Suncast.io and Ojos Ven); and allow journalists to curate the news (ShoutBoost), and skip to the part where the audience had left off (Catch Up, Resume).

As I previously mentioned, Hacking Journalism was the most unconventionally-planned hackathons (teams were assigned). It also produced some of the most creative hacks around video. Unlike Newstritious, Video Pizza's developers were not dominant. A developer I interviewed on June 6, 2015 even complained that this hackathon had felt "too much like work." Organizers hencouraged the journalists in the room to produce tools they could bring back to their workplace – a tall order for some teams. Video had become a big source of revenue for

newsrooms (Pew 2016: 55) and participants were invested in a realistic hack, with the exception of Ross and Amelia, who had enjoyed building useless hacks at "Stupid Shit No One Needs and Terrible Ideas" hackathon the latter had organized the year before. Ross and Amelia looked for inspiration in ideas completely unrelated to work. Amelia currently works as a director of innovation for a bank (Amelia Winger-Bearskin, e-mail to author, September 24, 2015), a testimony that one did not need to build for a market before they could secure their place in it.

Table 16	**Video Pizza's Distribution of Expertise**	**Skills**
*Amelia Winger-Bearskin	Opera singer, visual artist, coder, academic lecture and organizer of the Stupid Shit No One Needs & Terrible Ideas Hackathon at NYU. Later took on a job as Director of Innovation at a credit rating agency and now Director of the Interactive Digital Environments Alliance.	Front-end development
Liam Andrew	Student in Comparative Media Studies at MIT, STS researcher, software developer, and musician	Developer
Kristina Budelis	Videographer at The New Yorker. Co-founded of KitSplit – a marketplace for camera rentals backed by Hearst Media, six months before HJ NYC.	Videographer
Andrew Montalenti	Chief Technology Officer at Parsely, VA. Mentor and team leader.	Developer, Project management
Alessandra Villaamil	Mobile UX Designer and now Senior Product Designer at The New York Times. Had graduated from NYU's Interactive Telecommunications Program (ITP) two years prior.	Product Design/UX
*Sharon Chen	Interactive designer for Bloomberg News.	Design and front-end skills
*Steph Rymer	UX Designer at Moment – a design consultancy in NYC	Design/UX
Ross Goodwin	Student at NYU's ITP. Previously an economics major, working in politics. Recently held an exhibit on his camera that prints poems based on the facial features of the person.	Developer
Daniel McLaughlin	Developer at The Boston Globe. Ross's best friend from their college days at MIT.	Developer

The Video Pizza team had four developers, three designers, one videographer, and one person with additional front-end coding skills, for a total of nine members. The team ranks as extremely tech-savvy and their hack was a creative type. The team at Video Pizza came together at 8 pm on Day 1 of Hacking Journalism. Near the time Condé Nast closed for the night, the organizers had convened all teams and invited them to share their needs. Video Pizza was born then. Spearheaded by former hackathon organizer Amelia, who did not like the chauvinist attitude of her designated team leader, four women had decided to leave Team 8 (untitled) that night and merged with Andrew's team. Coincidentally, Amelia was a friend and classmate of Ross's at New York University's Interactive Telecommunications Program (ITP) in the second team. Ross was also connected to Daniel McLaughlin, who like Jihii was another repeat from the hackathon in Boston.

As it turns out, both original teams were interested in slicing videos. Andrew, the designated leader for team Video Pizza and also a mentor, paired up with Kristina to organize all eight of their members' ideas. Andrew quickly established a protocol for taking in feedback, but was open to letting the other members run the show. The five women on the team had a good relationship and used humor to integrate each other's ideas. The team decided to write their ideas on sticky notes that they stuck on the white board for the team to re-organize in themes. The team remained undecided for a while.

In Example 7, Ross pushed the team to think big, interjecting in the middle of a brainstorm that doing anything linear was not exciting enough of a concept (lines 1-2). Kristina, who was co-leading the session with Andrew, acquiesced in acknowledgment (line 3). Ross tried to address the diverse interests of the group (line 4 again, "even if we don't like the explosion of ideas"), but he clearly was interested in his friend's idea of forking paths (line 5).

Example 7

1	Ross:	I just don't think that doing anything with a linear timeline is super
2		interesting.
3	Kristina:	yeah.
4	Ross:	***let's*** embrace the non-linearity. Even if we don't like the explosion of
5		ideas, Dan's idea of forking paths. I think that a linear timeline with
6		extra options that pop up is just ***too*** limited for what we've been talking
7		about.
8	Kristina:	right. *((writing on the wall "non-linear timeline"))*
9	Daniel:	We could have a through line and have come out of that sorts of detours
10	Kristina:	sort of like that *((points to a drawing])*
11	Daniel:	no.
12	Kristina:	*((starts drawing))*

In Example 8, Daniel described his idea of a cool hack. A conflict was apparent from the rest of the transcript, as Kristina was amazed by Dan's proposition and ridiculed it (elsewhere, "why would you want ***that***?").

Example 8

1	Daniel:	you sort of leave this line and jump back like it's a straight thing and it
2		moves you backward… or forward… or *you could* have all sorts of weird
3		things where it *feels* like a straight line but actually they're little kind of
4		like sliding platforms. You get on this thing, and suddenly you're
5		somewhere else on the timeline
6	Kristina:	*((stopped drawing))* how would you draw ***that***?
7	Amanda:	like ***that***
8	Kristina	*((scoffs))*
9	Daniel:	I don't know.
10	Amanda:	it zips like that *((makes the gesture and looks at Daniel for*
11		*confirmation))*

The journalists on the Video Pizza team had trouble visualizing Daniel's hack. Video Pizza was trying to find the best way to play with the linearity of the YouTube timeline, which could offer a more flexible narrative once the segments were cut and pasted together for viewing. The team employed a lot of imagery to illustrate their concepts; to the less tech-savvy, Ross and Daniel's ideas were difficult to understand given that its value was technical rather than social. An explosion referred to how many pixels were changing in a video segment. If one could think

of movie files as compressed images, one could go from 3D to 2D to a line, to an array of lines that would have information of rgv value. Each pixel can have a value from 0 to 1 for each of the 1s (for instance, purple has an rgv value of 101.) The more pixels are changing in the segment, the more likely it is that the content in that segment is interesting.

Non-linearity was also significant from a technical standpoint. Non-linearity was a paradigm in neural networks; deep learning techniques were increasingly used to automate the news in 2016 (Johri 2016). Ross and Daniel envisioned combining just two inputs, but computing from multiple unknowns. It was a non-destructive way to edit, capable of ingesting video and audio feeds when a developer wrote the code to append metadata to a clip. Ross was pushing the team to "embrace" the possibility of making something quite viable in the future. Non-linearity was synonymous to the cut and paste method, but instead of watching a video from start to finish, a program would "generalize" its content and create a decision list to make it easier to extract and export any segments of the video to another tool. One could "deep dive" into content, and "slide between platforms" to end up anywhere on the timeline (lines 1-5).

Given that Video Pizza was a team with a history of not reaching consensus, new tactics emerged. Some women preferred to hang out and entertain each other while waiting to demo. The originality of the hack kept the programmers very engaged in brainstorming the software resources it would need, but when their idea failed to stick, they remained in their corner quietly coding in the last hours. The only source of engagement for the journalists on the team came from exchanging their past experiences and taking the hack back to their respective workplaces later. The videographers tended to prioritize tools that they could use at work, surpassing in majority the developers' interest in a creative hack.

In the few hours they had left, Ross and Daniel decided to get started but were left

dissatisfied. They continued working on it outside the hackathon. The developers by then had settled on identifying stop words to index the most important segments of the video in lieu of an explosion of color. Despite their differences, the product reflected the compromises the developers had to make.

Chapter 4 Conclusion

The examples of the Newstritious and Video Pizza teams at Hacking Journalism illustrated developer leadership in the production of the hacks – a more disciplined form of intellectual labor that was engaging as long as its steps could be translated to appeal to journalists. Ying and Alex's developer-centric management style had a profound impact on the Newstritious collaboration. Ying was able to relate a code problem to a major issue in data journalism and used his energy to bring his ideas home. By contrast, Alex had a soothing presence on the team, and used his procedural knowledge of coding to teach the journalists how to visualize the different steps to building the hack. The incommensurability of Ying's mobile experience and Kyle's design experience at a given moment can be explained by their work practices. Newstritious was a Chrome Extension, and therefore required extensive knowledge of Javascript, which a mobile developer like Ying only used sporadically. Journalists could listen to a developer, but the developers not so much to one another; different programmer's subjectivities made it especially difficult for two developers to understand each other. Chapter 6 later will cover the breadth of these coding practices.

Being able to map out journalistic practices and align them with a developer's objectives was not always feasible. Contrary to Newstritious' developer-centric style of leadership, the Video Pizza journalists could not open themselves up to sophisticated developer concepts (i.e. Daniel's explosion of images). Non-linearity, a chaotic process in which unforecasted news

could happen, preoccupied Daniel of *The Boston Globe*, who drew from the metaphor of the fork function – a practice in software development – to explain the value of the explosion. His use of technical jargon created resistance around his ideas, and reveals a clash of aesthetics between two classes of workers. The journalists had not been exposed to the inner logic of technologies that were taking the newsroom by storm in 2016, and as such, were unfamiliar with the material basis of its assymmetrical production of social relations in the newsroom. This suggests that journalists did not often work alongside developers; *The New York Times*, for example, had its journalists come down a flight of stairs to talk to its developers on a separate floor – an insight that was brought up at the video journalism hackathon, which may or may not be true in 2017. In the case of Video Pizza, the radical politics of the developers failed to be taken up by its interlocutors for failing to address the things of professional value to videographers – those ultimately with real authority in the production of the cultural content of video journalism.

The thematic medium – video – nevertheless afforded the teams in New York City more leeway to be spectacular. Many teams used advanced machine learning techniques to map and transform speech and video, and produced elegant, multisensorial demos. The journalism industry embraced developments in data science and mobile development to ensure their livelihoods, and many came together at hackathons because they were not only learning how to talk to developers, but some even coded now, and were readily engaging in forms of immaterial labor. It made for some of the most creative hacks at journalism hackathons because the demographic was exceptionally tech-savvy.

Section III: Deutero Learning

This final section of this dissertation establishes the significance of learning how to learn in disciplining play labor. Chapter 5 shows how two teams disciplined their play to facilitate this process. Teams that were successful in building their hack on time, or came up with a creative solution last-minute, were different from the teams that struggled to build a hack of interest from their ability to learn how to learn from intelligent systems. Chapter 6 shows how this ability stems from the varieties of hacker literacies in a heterogeneous team setting, which confirms some of my preliminary findings in Chapter 1, where I used the organizer interviews, judges' comments, and post-hackathon updates to the hacks, to argue that what hackathons are really about is the labor-power of these teams. The participants' plural acts of valuation judged not the merit of a hack, but their team's.

Learning how to learn is a concept pioneered by Gregory Bateson (1972) and based on his reading of information theory, particularly Claude Shannon's (1948) "Schematic Account of a General Communication System." Bateson described deutero-learning as a playful process similar to his model of metacommunication. Table 17 describes the stages of learning to lead to deutero learning. I am interested in the jump from Level II to III in the metacognitive processes he identified in Table 17 below: how participants were able to plan ahead because they learned the underlying theory behind any utterance or code. Instead of a simple corrective change in how the sequence of experience is punctuated, how does one correct errors systematically? Hutchins (1995) later argued that reasoning is embodied, and that social practices determine what mental representations of the changes in the system get produced. Such a diversion from Bateson is nevertheless key to understanding how the experience of programmers in communities of practices allows them to develop this ability to learn how to learn from specific sociocultural

work practices.

Table 17 - Bateson's Tiered System of Deutero-Learning

Learning IV	'…would be *change in Learning III*, but probably does not occur in any adult living organism on this earth.'
Learning III	…is *change in the process of Learning II*, e.g. a corrective change in the system of *sets* of alternatives from which choice is made.
Learning II	…is *change in the process of Learning I*, e.g. a corrective change in the set of alternatives from which choice is made, or it is a change in how the sequence of experience is punctuated.
Learning I	…is *change in specificity of response* by correction of errors of choice within a set of alternatives.
Learning 0	…is characterised by *specificity of response,* which – right or wrong - is not subject to correction.

Table 1: The levels of learning

Source: Tosay 2006, 3.

This turn to sociocultural theory brings me to a second proposition: that this collective learning process is sedimented in participants' hacker literacies (Santo 2011). As both critical and participatory media literacies, hacker literacies allowed hackathon participants to understand inherent biases in the design of their hack. This assumption is based on Castells' (1996) theory that those who possess skills in programming also had the education and ability to re-train themselves. The following chapters confirm his thesis in participants' use of deutero-learning – a process to which Castells alluded but did not capture ethnographically – as a kind of disciplined play that gives some tech-savvy participants a strategic advantage over others.

Chapter 5 – Disciplined Play

Disciplined play is perhaps the noblest and most valuable form of intellectual labor among hacker teams. This chapter describes one of its forms. No matter how engaging affective labor and spectacle were at hackathons, productive play, which I define as play that is focused on production and not so much on managing the collaboration, lent itself to hackers showing much evidence that they were learning how to learn by mirroring each other. In this chapter, I compare two teams, LIBOR Alt +Repair and Sirona Care, whose use of game simulation and paired programming session benefited from the complementary homogeneity of their team composition to enforce disciplined play. The examples below show that this higher form of play labor was of strategic value to hacking.

LIBOR Alt + Repair's Simulation Game

The LIBOR Alt + Repair hack was the best use of simulation to engage its team of hackers in problem-solving. The hack offered an alternative benchmark rate less susceptible to manipulation than LIBOR. In Example 1, their leader Kate started to model one of "a couple best-solutions" (line 1) for securing access to market data for calculating this alternative rate. They dug through yesterday's dataset, and projected their own rates from the raw mock data.

Example 1

1	Kate:	So, I think we have a couple of second-best solutions. One thing
2		we can do, is we can play some what-if games, right?
3	Naushard:	You can get the negative comments
4	Kate:	Yeah, I can get the sum of this but I want James to…
5	Naushard:	do some game theory stuff
6	Kate:	Do your game theory magic *((laughs))*
7	Naushard:	Right.

In Example 1 Kate and Naushard shared a joke; we are still in the realm of affective labor in political action. But I want to draw attention to the "what-if" games to which Silz-Carson

alluded in line 2, which involved exposing the team to the algorithmic thinking of some financial tools. Naushard suggested then that they get "the negative comments" (line 3). Responding to that suggestion, Kate confirmed that she would be able to get a sum out of these [negative comments], but wanted to enlist James' help on another matter, possibly concerning another data type (line 4). Naushard's proposition expressed the latter's aptitude for discerning the structure of financial markets, as well as the uses of specific data types e.g. negative comments for calculating LIBOR, which Kate shared as evidenced by the way she approved of the use of negative comments to produce useful data. But the professor delayed the motion and invited the rest of the team to play her game. The game is another form of play labor, drawing on constructivist pedagogical methods to communicate the rules of LIBOR.

Example 1 demonstrated how Silz-Carson deployed the expertise of several team members, namely James and Naushard, to accomplish a common goal: exploring and settling on second-best solutions. In line 5, Naushard advanced his interest in pursuing game theory. Responding to his enthusiasm for the subject, Kate repeated his statement and jokingly embellished it by calling it "magic" in line 6. Kate situated game theory within the latter's domain of expertise and possibly the element of spectacle in their hack.

Example 2 then fully shows how the team practiced a kind of Piagetian play – manipulating symbols and constructs to teach the mathematical thinking behind LIBOR. This was invited in part by Quan's suggestion in a previous example: "if we can compare like, the no-factor, the unknown one, we can see whether they're doing LIBOR or not." In lines 8-9, Kate provides a meta-commentary on her actions to teach her teammates how to calculate LIBOR from raw data. She thought out loud that she should "play with different calculations" and emulated a group of bankers, whom she assumed would want to "move things to change [the

LIBOR rate]." One bank would apparently forego, and another would also forego. As she talked over her calculations, she wrote out an average of 5.085. The group participated in her calculations, echoing her results to show her what they learned from her actions. Quan expressed his confusion when he exclaimed that the numbers she made up – six through twelve, appeared "so:: random" (line 5). Naushard frequently interrupted Kate's think-aloud for a clarification (line 12, "but if you take the median"), then providing a result (line 16, "it would be the same [rate as the average of 5.085]") and finally repeating the outcome (line 18, "regardless"). At this point, we can conclude that he had fully reasoned with her solution. By solving a math problem, Kate modeled how bankers arrived at the rates, which the team pooled to form the alternative rates.

Example 2

1	Kate:	We've got 16 banks. That's right. Let's play a game here. Are you
2		on this document? I'm on page 9. *((takes out a white sheet of paper))*
3		Put a new piece here. So let's play with the median. We've got 5.01.,
4		5.02, ….04, .05
5	Quan:	6, 7, 8, 9, 10, 11, 12. It's so:: random.
6	Kate:	13, 14, 15, 16. So these are 16 submits. The key with these numerical
7		games, is just you kind of want to find a plausible set of rates that are,
8		um… And you want to play with different calculations and say "okay
9		how much does a single banker, or group of banks, have to move things
10		to change?" So under this current system, those forego, those forego, and
11		and these guys get average, right? So the average is 5.085. *((laughs))*
12	Naushard:	But if you take the median- 13 Kate: In this case,
14	Naushard:	//then it would 5.08
15	Kate:	no, the median would be the average of these middle two
16	Naushard:	so it would the same.
17	Kate:	So the median would be 5.085 regardless
18	Naushard:	regardless.
19	Kate:	regardless of whether you drop the top and bottom or not.

Given the labor-intensive nature of manually mining data that was not available, the team's homogeneous composition was appropriate to seek out alternative datasets. The low

technicality of it required less coding and more calculations by hand. From this play labor, simulation helped the team forge a path for future diversions. The team would need, first, to get acquainted with the logic of LIBOR.

Paired programming with the Sirona Care Android team

Paired programming, which involves two programmers coding together, was a working arrangement between the two members of the Sirona Care team when they had difficulty merging their code. The paired programming session with Sirona Care fulfilled a second objective: to mentor a novice at Java with the same aim to troubleshoot poorly written code. Example 3 demonstrates how Katrine and Priya co-wrote the code for an app emulator.

Example 3

1	Priya:	Something slightly less terrible, we're only going to have two variables,
2		one of them is gonna be a random from the list defined in the variable, and
3		we're going to run that five times. Slightly less bad, less than happy?
4	Katrine:	Uh…Yes.
5	Priya:	Right?
6	Katrine:	Less than happy.
7	Priya:	So, for n_tie equals 0, uh i is less than equal to... Do this 20 times.
8		And then… In this parameter, we're going to pick a random variable.
9		So... ranheartrate, we're going to print that, and ranheartrate is an int!
10	Katrine:	Random.dot
11	Priya:	We're going to have this list. Intervals equals… uhm. Sum heartrates
12		equal and minus 2. Intervals. Array.dot as-
13	Katrine:	[Uh-hm. Arrays.dot
14	Priya:	And so, from those, this is going to pick(.)
15	Katrine:	Which one did you want? For random number
16	Priya:	So int, u::m index of hr, is going to be(.)
17	Katrine:	random x. And this is like a range, we're going from 0 to 4.
18	Priya:	Is it x comma?
19	Katrine:	x minus 10.
20	Priya:	wait
21	Katrine:	plus. Oh, 'cause it adds up
22	Priya:	O::h, so the max is going to be the length of this, you want to pick one of
23		these coordinates. sumhr.dot
24	Katrine:	… Oops oh yeah, because now we're pulling a new range.
25	Priya:	But that's okay, (.) because we're creating an index between 1 and 5.
26	Katrine:	u:::m… random().dot. Open parentheses.

27	Priya :	So this is the index of hr. We're going to run that 20 times. It's going to
28		create this list. And now we have a new variable.
29	Katrine:	You don't have to ask for that.
30	Priya:	This is the n. This is posting.
20	Katrine:	Oh (.) so we're missing a bracket up. Is this what's going to happen?

The transcript captures the instructions that Priya recited to Katrine as the latter wrote a program to illustrate a theoretical change in a user persona's heart rate on her mobile phone screen. An algorithm lists set of operations to be performed step by step, and solves a class of problems. A computer program is a sequence of instructions that comply with the rules of a specific programming language, and is written to perform a specific task on the computer. Priya was writing the functions (or operations) used to implement the algorithm along with their program written in Java to display a user's heart rate on his Android phone.

In lines 1-3, Priya told Katrine what she intended to do: name two variables and call on one of them to pick randomly from a set of numbers. Then, in a manner of inviting Katrine to follow her, she called it a "slightly less bad" alternative, which Katrine must vet as "less than happy" (line 3) before the former can proceed. Katrine took a couple of seconds to decide, ultimately settling on that option (line 6, "less than happy"). Priya named all the variables, and knew right away to associate the integer *n* with a tie class to make up *n_tie* (line 7). Tie classes can restore the functionality of some pieces of code in places where Java does not support multiple implementation inheritance. When she assigned the value zero to n_tie (line 7, "for n_tie equals 0"), Priya declared the location of *n_tie* to have an address of "0" in the memory of the computer, so that when the program ran, it reserved a space for the variable at that address. *N_tie* was the first byte, set to zero. Next, Priya initialized an index variable *i* to some value; the *for* loop required that she specified the number of times to repeat certain statements in the governing block of code, in that case, $i \leq 20$. This parameter instructed the computer to look up

twenty random values before it computed a heart rate (lines 11-12 “In this parameter, we’re going to pick a random variable... So, ranheartrate, we’re going to print that”). She added that *ranheartrate* was an integer (line 9, “ranheartrate is an int!”) – representing a binary value, and one row in a table. At that point, Priya read the line of code out loud “sum heartrates equal and minus 2” (lines 11-12), which calculated *ranheartrate* by summing up the random numbers and subtracting two on both sides. It is important to keep in mind that looping is important for compiling data in large databases.

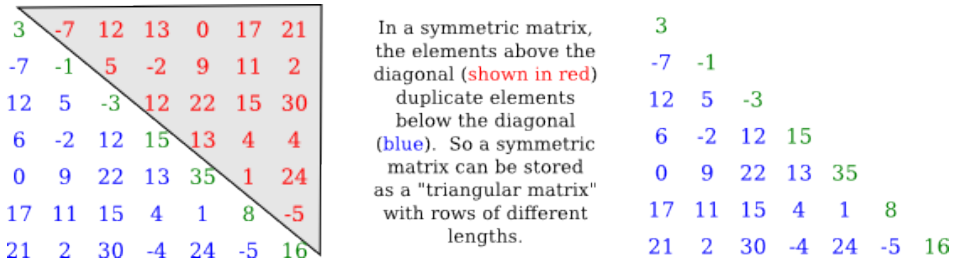

Figure 21 – The Logic of Memory Location

Katrine eventually picked up where Priya left off. In line 11, Priya was trying to declare the list of intervals at which heart rate data would be collected on the user’s phone. Seeing that she was still thinking, Katrine probed further (line 15, “which one did you want? For random number”) and took her laptop. Priya ended up naming it *indexhr*, with Katrine continuing the same work, reading aloud the code as she specified a range of 0 to 4 for *indexhr* (line 17). Even when she stopped coding, Priya re-read Katrine’s syntax and sometimes caught the latter’s mistake (line 18, “is it x comma?”), which helped Katrine correct herself (line 21 “plus. Oh ‘cause it adds up.) Their dance eventually contributed to their discovery of the internal rules of Java programming. Priya and Katrine’s commentary offers important insights into their collaborative learning process: the girls were supposed to create a new index to pull up a new range for a second array of which they needed to figure out the maximum length. Priya recommended using *sumhr* as one of the coordinates to have the program point to the memory

location of that second variable (lines 22-23, “O::h, so the max is going to be the length of this, you want to pick up one of these coordinates, sumhr). They therefore needed to declare *sumhr*, briefly deliberating over its conditions. In line 24, Katrine articulated the assumption that she was now undoing (“oops oh yeah, because we’re pulling a new range”). Priya echoed that thought, by okaying the procedure on grounds that they would be “creating an index between 1 and 5” to make up for it. It is not always advisable to operate without knowledge of the constitution of a new array, but they managed in lines 20-28, ultimately creating a new variable (line 28 “we now have this variable”). Katrine then proceeded to correct her syntax herself (line 31, “Oh (.) so we’re missing a bracket up.”). What was special about these two programmers was that they possessed extensive theoretical knowledge of computer science to locate data in memory and compute unknowns. Katrine nevertheless expressed wonder at the outcome of their project (line 31, “Is this what’s going to happen?”) when Priya showed her that the variable *n* was posting.

This example illustrates perfectly what hacking is about: setting up an experiment. Katrine and Priya were not experts in Java programming, but they helped each other figure out its rules faster by pairing up on an experiment to loop variables in their first program. By keeping up with each other, thinking out loud and self-correcting, these girls taught each other *where* to spot mistakes and together made up a rather flat hierarchy in the dialectic process of learning how to code next the machine.

Chapter 5 Conclusion

This chapter’s goal was to set the stage for the next chapter. The two kinds of play labor between the two teams – paired programming and simulation – have in common their

homogenous, semi-expert composition. Possessing deep knowledge of a domain is a necessary prerequisite for learning concepts faster.

However much of a novice Quan was in terms of LIBOR, the Alt + Repair team was engaged in a pedagogical technique called "what-if" games that immersed its members in the mathematical thinking behind LIBOR calculations. One only needed to understand arithmetic to participate. But to continue, their team leader had recourse to disciplinary tactics in economics. Using a formula that seemed dubiously simple, Silz-Carson then walked the team through the production a set of "plausible" rates that she based on a set of agreed-upon conventions. I call this process of exploration "disciplined play" because it requires carrying on a repetitive task to produce different outcomes and divert from the traditional logic behind LIBOR.

Similarly, the girl-duo behind the Sirona Care emulator showed how programming required knowledge of other layers in the backend. Coding together, their think-alouds illustrated the process of hacking by setting up an experiment to explore their own program. Most programmers will explain that they have access to shortcuts in programming once they grasp the underlying theory behind the rules of a programming language. What was interesting about this pair was their ability to mirror each other's steps to figure out when to declare a new variable, whether it should be random and whether it should be an integer; this mimetic play, set with picking up where the other left off, facilitated the distribution of intellectual resources in this collaborative learning process where authority operated in a vacuum (none whatsoever). Paired programming has in fact been recommended for those new to programming, partly because solving problems in programming requires individuals to keep track of a lot of information at once.

Chapter 6 – Hacker Literacies

Chapter 5 demonstrated how mimetic play worked well for homogeneous teams to facilitate several forms of deutero-learning. What happens if one ends up with a heterogenous team? The case study of CampaignCon documents some of the same kinds of "disciplined play" we have seen with LIBOR Alt + Repair and Sirona Care, but each corresponds to various proficiencies in reading and writing code. I propose that individuals' hacker literacies – as knowledge of some of the issues associated with coding collaboratively and in more than one language – directly played into the kinds of strategies that heterogeneous teams like CampaignCon used to solve problems around their respective semi-proficiencies. This chapter also illustrates the diversity of programming practices, and the sociocultural aspects of hacking.

While I borrowed the term "hacker literacies" from Rafi Santo (2011), I use it differently. While Santo defined hacker literacies as "participatory practices" that reconfigure sociotechnical digital spaces in relation to power (1), I am very much interested in describing code varieties, and repertoires of programming tasks upon which members of the CampaignCon team drew their strategy for multi-language coding. The examples in this chapter are woven into a narrative; each scene denotes a speech situation (Hymes 1972). Ultimately, I show how hacker literacies disciplined the production of the Campaign Con hack, by giving them to the means to creatively navigate the multi-lingual codescape of programming. Unlike doctors, the authority of each team member shifted at different points in time, depending on the resources he or she used to address the needs that erupted from the technology.

Scene 1: Pitching, Brainstorming and Framing Information Literacy

From Bruce's pitch, several of his experiences as an activist and systems engineer are used to frame the importance of the hack in this genre. His reference to differentials suggested

that he knew exactly from which programming repertoire to draw. Negative totals also mattered in this specific assemble of campaign finance practices. But Campaign Con teammates did not always share his goals of creating a tool to visualize differences in campaign contributions and systematically analyze influence in presidential elections. His co-leader, Dhrumil, in fact had come to the hackathon wanting to learn how to audit a government database. To further draw in participants to his idea, Bruce used his activist voice to tell the story of a scandal to frame the importance of the hack.

> "Here's what it looked like in 1990, there was one person… look at all these people now who are, like, controlling that, you know, that first stage of elections… and here's… differentials that are happening… you know, people who are giving over the limit and all of a sudden, their contributions are getting distributed between their children or the wife, or… they're giving to another… reallocated to other PACs… or maybe they're refunded, too. This happens. This committee has to give money back… it's funny, one of these years in here, I can remember, there's actually two candidates that had negative totals, and basically, they must have received from, you know, this large donor-contributor but then refunded it within that cycle so… *((laughs))*, maybe they decided "maybe we don't want to take it from so and so, because, you know, he's a shady character *((laughs))*"

Activists from hackathons often humored each other by pretending to speak like a corrupt politician, here a literary trope used to differentiate among the values ("***this*** [the differentials] happens") that each of his teammates expressed during the brainstorm. Within the context of his work, however, campaign finance law forbids individual donations over $2,700 per candidate committee, but an influential individual wanting to exceed that limit could simply reallocate funds through other relatives or organizations to sponsor the same candidate, which could be detected by the "negative totals" that this practice produced. Bruce went on to work for Lawrence Lessig's Citizen Equality campaign, as part of the broader work that communities of practice performed around his founder, Professor Lessig, to advance campaign finance reform.

The first half of the brainstorm session concerned itself with framing of the value the hack in terms of such a political ecology. The Federal Elections Commission (FEC) is an independent

regulatory agency whose role is to oversee the public funding of federal elections and disclosure of campaign finance contributions. In Example 1, the new team of five established that the FEC lacked the capacity to audit its databases (lines 4-5, "they've been doing it for several years, but no archival?"). The activists on the team – all except for Perihan and Nathan, were familiar with political newsmaker Open Secrets, which had been acting as the leading auditor of campaign finances. According to Bruce, Open Secrets had tried to build the capacity for automating some of that work for years (line 3 "been doing it for several years now"). Upon hearing this insight, James expressed his surprise that neither agency had already built a system for storing and reading archives.

Example 1

1	Bruce:	I don't know, unless we start downloading a few of these files, it
2		could be as trivial as writing a simple script that says SQL import…
3		[Open Secrets] has been doing it for several years now
4	James:	I don't understand, they've been doing it for several years but no
5		archival?
6	Dhrumil:	***Never***.
7	James:	Oh.
8	Bruce:	They have all the years but I don't know how often these files are
9		getting modified.
10	Dhrumil:	Oh, but they don't have the weekly files
11	Bruce:	Right, they do the periodic thing… are they still modifying?
12	Dhrumil:	Well, he's saying "grab the 2008 files and 2012," right? So he's
13		modifying them between January and July 2012
14	James:	Ah, I didn't understand them.
15	Nathan:	They're taking from that database to make the visualizations and in
16		computer statistics, they're always adding to that database, that's
17		what I understood. I think it's a process of continued…refinement,
18		you know?
19	James:	It's a ***whole other*** ball-game.

As they formulated their pitch, Dhrumil pushed his teammates to explore the structure of the dataset before they decided on a path to follow. Upon inspecting it, the software engineer for a New York City political news agency noted that the 2008 and 2012 files were not updated

regularly (line 10, “oh, but they don’t have the weekly files”), prompting him to look at the dates when they were modified (line 13, “so he’s modifying them between January and July 2012”).

Scene 2: Division of Labor, Novices and Gurus

Example 2 is the start of their division of labor. Perihan had arrived late, at which point Bruce and Dhrumil, who had reportedly acted as the team’s de facto leaders from the get-go, debriefed her (line 2, “We’re just getting the diffs set up”).

Example 2

1	Perihan:	Sorry I’m late
2	Bruce:	it’s fine, we’re just getting the diffs set up.
3	Perihan:	Ya::h, that’s good.
4	Nathan:	I’m saying, we shouldn’t worry about graphics because we’re working
5		with the same dataset… We’re just showing the audience what data we
6		have, and you guys are doing something different with caching and
7		updating to try and build a bigger backend and we can add on these
8		different compartments for comparatives as needed but that’s just for the
9		audience to see “This is what we’re doing.”
10	Bruce:	Yeah, yeah.
11	Nathan:	So I think we need some magic impact from the video, if it could change
12		some slides, we’ll convert them to Christianity or something.

Even though participation was open to everyone, participants nevertheless had to legitimize their contributions by presenting themselves in the most capable light. The technical members of the team cannot follow Nathan’s verbiage in lines 4-9 (“you’re doing something different with caching” and “adding these different compartments for comparatives) because it only makes allusion to Bruce’s project, which at this stage is a separate component of the hack. As already evidenced by James’ previous evaluation of his earlier statement in Example 1 (line 19, “that’s a whole other- ball game”), Nathan had not yet mastered the discourses circulating in digital activism and presented himself as someone was had been exposed to programming language but did not code himself. No one, for instance, had previously mentioned caching or a backend. Bruce only suggested writing SQL script in line 1 of Example 1, which did not require

a backend, either.

The team danced around the possibility of switching to a more technical discussion, with Bruce, James, Nathan, and then Perihan, each making technical comments in their own time. Bruce might have owned the story for the hack, but Dhrumil owned its workflow. The data journalist had successfully redirected the brainstorm session towards a structural understanding of the FEC's workflow. He then got up to write a list of tasks on a white board:

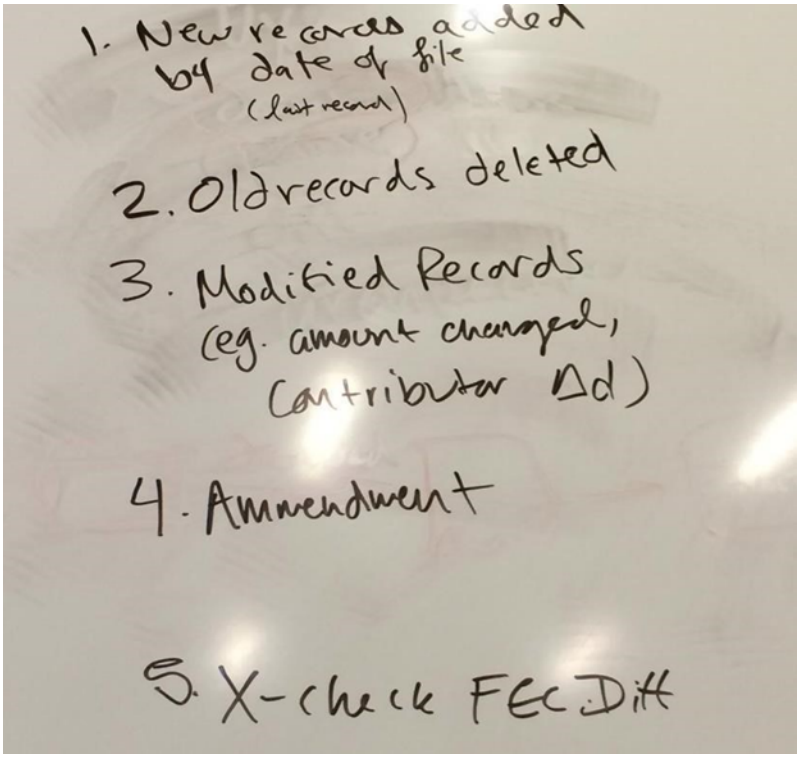

Figure 22 – Campaign Con Workflow

Figure 22 was a very rudimentary sketch of the team's division of labor, set to closely mirror the workflow of the FEC. The following modules were built to execute these tasks:

> **Models:** Using the Peewee Object-Relational Mapping (ORM) technique, this module handles converting data from incompatible type systems into the pythonic values used by the database. Models.py creates the tables in the database by first defining its format i.e. model classes (table), fields (columns), and instances (row) and otherwise sets the appropriate columns, indexes, sequences and foreign key constraints for converting the data. The goal is to retrieve and store the data without losing some of its properties.

Main: Connects to the database. Main.py summarizes all the tasks the CampaignCon program will perform. After importing all the necessary modules, running the code will connect the user to the database server, create the tables, and drop all the tables in the database. Once it's ingested new data and mapped them onto the corresponding tables, it will display the results of each subsequent query to the database.

Downloader: Checks the FEC website to find if a new filing has been posted. If there is a new filing, store it in /data (whose?) folder with new date. It runs the hash lib module, which implements the SHA algorithm to ensure that unique file objects are produced for the data it downloads. It takes a base input number and computes a hash value that becomes the public key for secure message digests.

Ingester: The meat of the app, pushes new data onto database. Ingester.py imports from the Models module the following fields: File; Contribution; ContributionChanges; and ContributionHistory.

- A regex request searches, removes and/or replaces file objects continuously
- Parses the FEC files for various dates it translates into the appropriate table rows
- Creates a dictionary or lists of buckets for various properties of the data that it will then assign to the table columns. e.g. Transaction id; committee id; amendment id, etc.
- While the ingester runs concurrently to the downloader, it will insert a row if it does not already exist.

It will also check rows for modifications "amendments" and add a ContributionHistory object if that is the case.

Diff: a HTML file displaying the differences computed from comparing two types of contribution objects.

Views: Checks for deleted rows and returns (ret = [] means "return from subroutine] a JSON encodable version of the diff objects

Final: Creates an empty dictionary for comparing sequences of lines of text and producing human-readable differences that it then appends to a diff HTML file.

Figure 23 – The Six Modules Written by Campaign Con
Source: Campaign Con Github Modules.

The team's Github repository gives us an idea of how different modules – files ending in .py – carry out these tasks (Figure 23). Models.py creates the rows and tables for the database. The main module connects to the database. The downloader executes all the tasks that Dhrumil outlined on Figure 22: it checks whether new records have been added to the website, downloads them via the Ingester.py module, and deletes old records. Diffs.html calculates the differences in

amounts initially reported by donors in modified records (step 3), and views.py displays the amendments.

The following division of labor occurred after the brainstorm, and changed by Day 2. Dhrumil had volunteered to do the diffs and created a SQL Lite database the evening of Day 1. During the hackathon, Dhrumil mostly managed other team members and worked with Bruce to conduct queries on the data that they had been given as a test case. Having presented his data visualization project early during the brainstorm session, Bruce was hard at work trying to scale it: he mapped the network of donors, and the changes to their donation amounts, highlighting nodes that were related to each other, and seeing that some nodes representing one family could have redistributed his donations to a campaign by spreading the funds over several family members to beat the campaign donation limit. Perihan agreed to code to retrieve FEC files in Java, and had Dhrumil and the latter's classmate Al build around her code with higher-level control and customization in Python. Lower-level languages like Java are said to be closer to machine language, which is not human-readable. The difference with higher-level languages like Python is that the latter had simpler syntax rules. Python did not require code to be recompiled, and it could easily be implemented with other languages. Perihan wrote the code in Java to read files, download them, and delete old ones. She ended up coding most of the hack because after magically succeeding in reading a file, she was in a better position to figure out how to download or delete files. Nathan and Max stayed throughout the collaboration but could not find anything to do. Max's parser never made it into the design of CampaignCon and Nathan could not code. Later on Day 2, Nathan recruited Max's help in geographically mapping donors but this extension could not be built in time.

While Nathan might not have been able to participate fully, the choice of SQL appealed

to Dhrumil. In Example 3, he offered to do anything SQL-related (line 15), at which point Bruce playfully invited him to identify himself as the SQL expert on their team (line 16, "are you a SQL guru?"). The two programmers had in fact been instructing each other on the uses and possibilities of SQLite. MySQL is a database management system. Programmers use "Query Structured Language" (SQL) to run queries to add, access and store data from a MySQL server. In lines 3-4, Bruce called on Dhrumil to help write a query to ascertain what data type *amendmentID* might be. Bruce became engrossed in the task, exhibiting a certain ability to learn how to learn from his immaterial labor when he explained to Dhrumil that he was trying to understand how to pull two tables from one table (rows in a database), and later professed that he was still "actually wrapping [his] head around pulling" them (lines 6-7). In lines 13-14, he proposed a few options for comparing two tables.

Example 3

1	Bruce:	Oh, there's an *amendmentID* too. Didn't notice that…
2	Dhrumil:	Doing a query?
3	Bruce:	No, there's a separate thing called *amendmentID* so that might not
4		be a report type, so we should do a quick query on that
5	Dhrumil:	Okay. ah… which one are you working on, by the way?
6	Bruce:	So, I'm actually also wrapping my head around pulling… essentially
7		two tables from one table
8	Dhrumil:	So, … oh yeah, yeah...
9	Bruce:	Figure out (.) so then this would be
10	Dhrumil:	We can just like, (.) start...
11	Bruce:	Well…
12	Dhrumil:	No you're not-
13	Bruce:	depending on… so (.) I guess there's a few options for comparing two
14		tables
15	Dhrumil:	I can definitely with anything SQL-related.
16	Bruce:	Are you a SQL guru?

Scene 3: Non-unique IDs, mistakes and delegating

The Campaign Con team spent a long time testing their assumptions about its architecture. Dhrumil wanted to know whether the dataset was indeed circulated in Comma-Separated Values, a common format that makes it easy to exchange data by displaying information in a table-structured format (numbers and text) and plain text. This distinction proved important for everyone on the team, as Comma-Separated Values needed commas to separate each field. Dhrumil relayed this specification when he consented to having Perihan code in Java, suggesting that he was familiar enough with how Python worked with these formats to know how they all worked together. It was often the case that one software engineer coding in more than one programming language often learned the hard way *when* punctuation and capitalization mattered – in which language. CSV files are the start of a labor-intensive transformation of Javascript Object Notation (JSON) to the open-standard format that humans need to read data objects on the one hand, and make it easy for machines to parse and generate text on the other. The diffs coded by Dhrumil would be based on the JSON file itself, as this code snippet shows below:

```
from peewee import * from app import db
from models import File, Contribution, ContributionChanges, ContributionHistory

import csv import json diffs = [] diffs.append (diff)
diffs = filter(lambda x: x!=[], diffs)  << Selected from final.py
from datadiff import diff
```

Figure 24 – Dhrumil's Code to Import and Diff Campaign Contribution Files
Source: Campaign Con Github Repository.

Basically, Dhrumil's code implemented the peewee import library, then the Models module to help it import the FEC file and three tables with contribution data in CSV and JSON formats.

In Example 3, Dhrumil and Bruce led the inspection of the dataset. Dhrumil then proposed that they manually dug through the files on Excel. The team wanted to know if they could verify recent changes as they were made to the amended FEC files, and how frequently they were modified. In line 1 of Example 3, Dhrumil directed attention to the presence of a certain *amendmentID* class. Rather than simply assuming what it was, Dhrumil read such information critically. It was then that he discovered that another data type, *transaction ID*, was not a unique identifier (Example 4, line 4). A query on *transactionID* pulled 108 records (line 2).

Example 4

1	Dhrumil:	after the cop joke, I don't think *transactionID* is a unique ID. I got one ID
2		and it's got 108 records
3	Bruce:	Oh NO::O! *((grunt))*
4	Dhrumil:	So I don't know what the hell is *transactionID* is, but it's not a unique
5		identifier *((laughs))*
6	Bruce:	*((groans))* that means my query is probably total garbage
7	Dhrumil:	yah
8	Bruce:	no::o… what are we going to do? They don't have any unique identifier
9	Dhrumil:	what is *subID*?
10	Bruce:	*((mumbles for 30 seconds)...* save it to be safe
11	Dhrumil:	I guess they don't have the *transactionID* in here either, oh *tran-ID*!
12		"only valid for electronic files apparently associated with each itemization
13		or transaction period if you have a PC electronic file. It sounds like it
14		should be a unique ID.
15	Bruce:	well, only a percentage though. 'Cause they're not all electronic
16	Dhrumil:	o::h okay. Then why put anything there? Huh?
17	Bruce:	they had *otherID*.
18	Dhrumil:	*otherID* is… other contribution from other individual… this column is
19		null. So that's not useful for us.
20	Bruce:	*((groans))*
21	Dhrumil:	Okay, it probably won't mess with your query. Let's find out, 'cause in the
22		end, any transaction id that starts with an –s you can ignore. I don't think
23		this is good what we're doing, I need to find something else to do.
24		*((laughs))*

This mistake, which cost Bruce a full day of running queries was based on the assumption that all IDs were unique – a requirement to run SQL. Both Bruce and Dhrumil were shown in

Example 4 to at least possess knowledge of this constraint, which they used in anticipation of further roadblocks. Dhrumil reread the documentation looking for what other classes of ids did e.g. *subID* and *otherID*, and what variables would collude with them. Dhrumil directed Bruce to a line in the documentation that stated that the class *tranID* was only valid for the transaction period associated with the creation of the electronic file, and used it to reinforce the claim that it would produce single items (lines 13-14 "it sounds like it should be a unique ID."). Bruce disagreed in the next line, by stating that this was true of "only a percentage." The latter inquired into another class of id, *otherID*, which they both hoped would be a unique identifier. Dhrumil looked it up, and called it less than useful (lines 18-19, "this column is null"). As Bruce playfully grumbled that his "query is probably total garbage," Dhrumil weakly reassured him (line 21, "it probably won't mess with your query") and followed up with a contradictory statement (line 23, "I need to find something else to do"), suggesting that the hack had truly lost its value. Dhrumil sheepishly offered to write an algorithm, a winning trope for many hacks, but in his view, it was cheating.

Several CampaignCon team members had enough expertise to tell each other what coding literacies they needed, a point that Dhrumil illustrated shortly after breakfast on Day 2, when he commented that they were too low on manpower to work on Bruce's website: "I think we kind of are lacking in manpower. If we had someone to do front-end stuff, then we might be able to do the crazy website and have something to show." What they needed was someone with graphic design skills to put the visualizations on a website, a task that Dhrumil eventually took upon himself when he found a mentor, Ali. James initially had reservations about making a website, first citing lack of confidence that he had the requisite knowledge to do it all by himself, then emphasizing that their proof of concept would not suffer if they did not build it.

Scene 4: Learning FTP

It was not until close to 11 am the next day that Perihan finally got her program to read files. The programmer had spent the majority of Day 1 trying to execute a file transfer.

Perihan had struggled to write a program that could query the FEC's servers for campaign contribution data, for which a *fetch* command sends that information to the client server via a FTP (File Transfer Protocol, see Chapter 2). Before the user could read the data on his machine in a human-readable format, a FTP Client had the capacity to read the CSV file headers from a host server before it downloaded the data in a JSON format. This critical step would 1) present the data in the format the client used, and 2) avoid having to buffer a computer's memory stream before it *commits* the data into the database of the client server. When she exclaimed that she had managed to read a file, Perihan had landed on this script:

```
FTPClient ftpClient = new FTPClient();     try {

        ftpClient.connect(server, port); ftpClient.login(user, pass);
        ftpClient.enterLocalPassiveMode();
        ftpClient.setFileType(FTP.BINARY_FILE_TYPE);

Integrated with Dhrumil's Python code:

with db.transaction():

        myfile, _ = File.get_or_create(

            name = os.path.basename(filepath),
            years=next(re.finditer(r'\d{4}_\d{4}', os.path.basename(filepath))), sha1 =
sha1OfFile(filepath),
            updated = dateparse(os.path.dirname(filepath).split("/")[-1].
replace("downloaded_", "").replace("_", "-")).date(),
        ingested = False

        )
```

Figure 25 – Code to Replace Downloaded Files
Source: CampaignCon Github Repository

Example 5

1	Perihan:	Yes! Oh, I read it!
2	Bruce:	***Heyy***
3	Dhrumil:	What was wrong?
4	Perihan:	I ***don't*** know
5	Dhrumil:	*((laughs))*
6	Perihan:	I read it. I commit files from the FTP, in Java, and then I will make it, ah..
7		beautiful mySQL. So… um… I will read the, zip file names from the file,
8		download to the folder names, right now, I don't have time to write the
9		UI part, so I just wait for files and download to specific path. If you want,
10		I can ***add*** unzip. Also
11	Dhrumil:	So the main thing we need to do is
12	Perihan:	//is check
13	Dhrumil:	when you download, you need to check whether that file is already
14		there or not. So, only download it if it's a new file.
15	Perihan:	so… only download
16	Dhrumil:	so I guess what you need to do is download, unzip, and check
17		whether it's the same file or
18	Bruce:	They use the same file that-
19	Dhrumil:	//yeah, but there's a file(.) *hash*
20	Bruce:	*hash*
21	Perihan:	Yeah, but… I don't have that much time *((laughs))*
22	Bruce:	we can also store last date downloaded, they had that last update field
23	Perihan:	Okay, let… unzip part also, and then we can add another part
24	Bruce:	yeah.
25	Perihan:	I learned something because that's the first time I read something
26		from FTP.

In Example 4, Bruce and Dhrumil helped Perihan figure out whether she needed to use a hash to index every new file. Their gesture indicated that they could situate Perihan's code in the web of digital literacies in which they swam.

Additional literacy demands followed. Shortly after writing the data and control connections for the network protocol, Perihan announced that she had fixed the JSON control bug. JSON is a data interchange format that aims to be independent of the language in which the data was originally encoded. Once an attribute-value pair (e.g. Fruit: 'pear') is inserted in MySQL from a Python *json.dump()* function, it made it easier for the client to parse (i.e. analyze

the syntactic roles of different parts of a file, that is, read the header and commit the file) and record changes from the File Transfer Protocol (Example 4, line 6). Some of the encoded JSON strings, to which Perihan alluded, may have used "illegal" characters, such as the back-slash "\n" and "\t" to start a new line, thereby messing up the order of the display of the metadata. Depending on the framework used to encode information in numeric strings, the method for storage and transport between the encoder to the decoder on a web client would not be "binary-safe," making it difficult in some countries, for instance, to correctly display accents. The three programmers' obsession with hashes, commas and the like drew on their metapragmatics – knowledge of linguistic conventions to specific speech situations (Hymes 1972) within the historical coding environment and community, such as the use of symbols like "\", as well a metapragmatic awareness (Silverstein 2001) of the rules of the machine and strategic use of these rules for circulating information and encoding for transfer. I would like to point out however, that code being close to speech (Coleman 2012) makes the work of coding inherently playful. Perihan could run code successfully without knowing what she did (Example 4, line 4, "I don't know" [how I managed to read the file]). In other words, one could be proficient in a language and still do things with words (Austin 1962) that produced unintended consequences.

Scene 7: Mentoring new help

With only two hours left before their submission deadline, Dhrumil called on a former classmate at Northwestern to fix the problem of handling non-unique transaction IDs. As one recalls, SQL required Dhrumil to know the rules govern the relationships between data before he built the database; SQL did not allow for duplication of data. To curb these limitations, Ali suggested the team switch to Mongodb to have the flexibility of a dynamic schema, meaning that records could be created for unknown data types, or in this case, redundant fields or values. Its

rows could be re-ordered later.

Example 6

1	Dhrumil:	You're changing two lines of code? Oh, you got reconnected to ***Mongo***?
2	Ali:	some of it you wouldn't be able to change
3	Dhrumil:	yeah. *((turns to Bruce))* … He's pouring into Mongo db…apparently
4		with ***one line*** of code.
5	Bruce:	oh GOD…why didn't you do it ***in the first place***?
6	Dhrumil:	***I know, right?*** This thing's been AWFUL. Okay, cool so…
7	Ali:	*((mumbles))*
8	Dhrumil:	Class 3 files. Yeah. That was what? The other one was, our record being
9		deleted. That was interesting. Why is it being deleted? Yeah.
10	Ali:	( ) see the feedback loop?
11	Dhrumil:	and then maybe new records are- so are we bringing new records to add
12		in. Before (.) New records are obviously added after each round (.) Old
13		files and new records. That new stuff is not interesting but what is
14		interesting is the new stuff added before that date. Does that make sense?
15		So like being September 2014 and March 2015, they added a bunch of
16		2013 records. That's kind of ***interesting***…

While Examples 1-5 show that CampaignCon was able to negotiate goals for the next tasks, Example 6illustrates the variety (Dorian, 2010) of proficiencies Dhrumi, Ali, and Bruce possessed while finally cracking the problem of non-unique id's. Dhrumil shared his surprise at the relative ease in which Ali could program queries in Mongodb (line 1 "in two lines of code") with Bruce, who joked that he was appalled that Dhrumil had not thought of doing so earlier (line 5). The transcript suggests that the two programmers were aware that a certain competency was needed to solve the problem that they had encountered in Scene 2, and they agreed that Mongodb was a standard practice for solving it. The two programmers might not have used Mongodb in their everyday workflow, but they had been sufficiently exposed to its uses in data science to know that it was significant when it reduced the number of lines in their code (line 4).

Chapter 6 Conclusion

The paradox in coding brings me to the conclusion that Dhrumil and Bruce, despite not being experts in Java or Mongodb, had this ability to "learn how to learn" (Bateson, 1972). Learning how to learn entails managing uncertainty in a multilingual coding environment through a metapramatic awareness of the limits and possibilities in certain coding practices, giving some hackers a strategic advantage when seeking help and coaching others. Bruce and Dhrumil knew where to read for specific information about the architecture of the files (Scene 2), how to detect errors (Scene 5, "the transaction IDs, they not unique") and correct their course (Scenes 4, hashes).

Programmers have often told me that once one learned one programming language, it was easier to learn another. An acquaintance of mine explained that learning how to learn involved understanding the underlying theory behind a coding praxis, notably its communicative functions. Java is an older language than Python, used to program the Android operating system before Apple came out with a more mainstream operating system. Older engineers who attended a trade school were trained in Java, while the younger generation at MIT is currently trained in Python, like Dhrumil had been at Northwestern. Some programmers do not like Java because it is a very verbose language. However, and John Gumperz (2001) was keen to remark on linguistic repertoires – defined as the totality of linguistic forms regularly employed in a socially significant interaction, that the specific tasks Perihan could actually code relied on her knowledge of certain coding conventions in Java, which did not carry across Python and other programming languages. As silly as it sounds, these conventions differed only by their punctuation and syntax. Bruce and Dhrumil shared the same semantic map of coding regardless of programming languages, and understood Perihan's best practices for coding. They could all

understand the cultural significance of fewer lines of code, of non-unique IDs, documentation, despite not fully understanding how one's machine read the situation for which she had written instructions.

The case study of Campaign Con has demonstrated the nature of disciplined play labor for a fairly heterogeneous team. Its diverse team could proceed methodically, and learn how to build something they had never built before from the careful coordination of their hacking activities. Like assembling blocks but with a strategy, CampaignCon's play labor was not about figuring out but copying and pasting names without knowing their history or what they did. Likewise, Perihan cracked her own code and was not able to explain how she had managed to do it. Their play is not virtuous like that of Townhall because CampaignCon members never excelled at anything in the course of their collaboration. What they did, they furthermore did with an explicit goal in mind, not for its own sake. Dhrumil's ability to manage his teammates' immaterial labor describes the very kind of participant a hackathon needs, and the kind of entrepreneurial worker-subject that newsrooms and digital government seek.

Conclusion – What have we learned?

We have seen from the structure of hackathons that the prize system did not allow participants to claim any significant resource in real life aside from a small token of recognition. Workers were only merely asserting themselves (Virno 2000). They had to figure out how to channel their creative energies in the service of making the technology work for them and in that process, engaged each other in disciplined play (Huizinga 1938, Caillois 1958, and Sutton-Smith 1997), using their hacker literacies (Santo 2011) to learn how to learn (Bateson 1972) to solve problems.

This dissertation has sought to make a contribution to the literature on play labor (Terranova 2000) by presenting a typology of play labor with varying degrees of consequentiality. Unlike other forms of play labor on the web, pleasurable forms of labor were used to engage the digital worker class at hackathons; they did not produce commodities, but subjects (Virno 1996). It is still very possible to observe a subset of hackathons in several industries and draw out their connections to more tangible forms of immaterial labor (Lazzarato 1996). The hackathons in this dissertation offered a mirror into upcoming work practices in the production of multimedia news, management of chronic conditions, and in the field of civic tech; these events accelerated the speed of digital production to make these trends more obvious to participants. The ability to learn how to learn became central to differentiating between the types of play labor, establishing one group of digital workers as superior to others in terms of the skills that this digital economy *will* or already needs. Play is embedded in language that since the postmodern turn in anthropology has focused much more on the agency of individuals in cultural production. It took hackathons, temporarily sandboxed spaces and spectacles, to communicate the very values that will single out these few individuals based on merit.

I have tried to create a typology of play labor by providing snapshots of the different kinds that emerged at hackathons in separate industries. My process of differentiation took these hackers' ability to transgress specific technical constraints. Because code was written around a concrete set of work practices, and builds off standards already written for that technology, each industry came out to have distinct preferences for the technology they wanted to hack. They produced different types of play labor to respond to the demands of the machine and social cooperation. As the comparative case studies have shown, play labor required more commitment in the presence of others. At civic hackathons, which were backed by communities with a long history of campaign finance reform in the New England area, participants engaged in a more egalitarian form of play labor around the virtuous capital of teams.

The case studies of teams at civic hackathons suggest that their digital labor revolved around the problems of locating the feedback mechanism by which to automate citizen support within a set of commands or "verbs." The demand for content negotiation, documentation, beta testing, deployment and task reporting, had motivated activists to come to a hackathon to learn how to lead an open-source project. Journalists were much more concerned with acquiring metrics of attention and realized that to improve them, their labor-power resided less in writing content and more on editing videos and parsing speech online to measure publics' affective response to news. It had more to do with working alongside developers to translate their editorial judgment into a set of instructions, which fed into the design of a computer program that can "learn" how to make better recommendations. More slowly, physicians' jobs have been replaced by telehealth or wearables, but the outcomes of healthcare hackathons suggested a trend towards the commodification of diagnostic planning; ubiquitous computing was making it easier to track medication adherence, and provide support on-demand. These developments reflect greater

accountability on the part of medical doctors, but it also replaced some of their planning roles. Both journalism and healthcare hackathons involved more practical forms of play labor that were directly tied to capitalist modes of production. The automation of certain tasks, such as writing the news and diagnoses, had conceivably displaced doctors and journalists and given them an advisory, more creative role. It was now possible to imagine a class of physician-entrepreneurs. Activists also had to pick up new skills in digital production and data science to take advantage of new developments in e-government.

The types of play labor were: humor; geeking out; mistranslation; pivoting and role reversal, leading to progressively more disciplined forms, such as non-linearity, simulation, and paired programming. Creativity was associated with higher levels of affective labor and the distribution of authority in teams. A flexible division of labor, geeking out (Ito et al. 2009), and light humor were observed at civic hackathons where activists coded alongside developers, while more didactic styles of collaboration were observed at healthcare hackathon. Developers deferred to physicians but also filled in gaps in programming-related information for them, even taking a leadership role in journalism. The balance of affective and intellectual labor at hackathons was tricky in view of different relationships the hackathon entertained with capital from different industries. Healthcare hackathons are more "serious" and higher stakes, with cash prizes on the line and legal regulations on record, while civic hackathons are communal and subsequently facilitated more pleasurable forms of labor. Journalism hackathons were learning opportunities; the reputation of journalists as tech-savvy, creative problem-solvers resulted in more creative hacks around a medium.

It was also easier for homogeneous teams with complementary domain knowledge i.e. The Cuff Guys, LIBOR Alt + Repair, to be more disciplined in their play. While collaboration

inside heterogeneous teams like Reciprocall did produce a teaching moment, this initial observation was challenged by new arrangements between more like-minded individuals in Sirona Care, such as the switch to a paired programming setting consisting of Java programmers Priya and Katrine in the middle of a hacking session.

I dare not be so linear in my reflection of what I saw, but excess and extravagance are types of play labor in the service of *en-gage-ment* (Boon 1999; my emphasis). However temporarily, transgression and affective labor did produce new relations between subjects. Let us forget the fantasy of production – the algorithm or the prototype – and focus on the actual thing that is being made at a hackathon: a recursive public (Kelty 2005), concerned with the infrastructure and maintenance of their livelihoods, which then required collaboration with external partners to learn from them trends in the future of work. By adopting flexible divisions of labor and team compositions, the play labor of hackathon participants aligned with the values and relationships they wanted to safeguard: diagnostics would not replace the doctor's or the journalist's judgment; government could also open its data, activists could build platforms, but any of these gimmicks would not eradicate values like transparency. This dissertation shows how certain conditions of work were communicated - and resolved - through play as virtuous conduct, spectacular modes of expression, and the socialization of the new worker-subject.

Although hackathons do not incur any sort of lasting relationship or network benefit, this dissertation is an invitation to the Italian Autonomists, as Terranova has recently suggested from her current interest in biohacking (e-mail to author, January 25, 2017), to consider this non-commodified play labor as having achieved at the hackathon a status equally important to salaried work. In this vein, it is inaccurate to say hackathons are useless. Removed from its institutional milieu, play labor at hackathon lent itself to a total prestation of skills (Mauss 2007,

73); considering that most relationships built at the hackathon never left the circuit of the potlatch, the value was elsewhere: the affirmation of participants' labor-power was in their ability to learn how to learn. Participants learned new trends in the future of work and how to code technologies that would gain popularity in the next year. With sufficient ownership, drive and the right team, one could learn how to exploit one's team's play labor for one's own gain. Organizers especially increased the visibility of their startups, nonprofit organizations received assistance on current projects, and generous sponsors could eventually produce some investments, but no one was obligated.

In this spirit, I end this dissertation by reflecting on my own failures as an ethnographer. In the middle of my fieldwork, I started betting on teams, choosing those I felt would win a prize or finish building on time. I felt that if I focused on the winners, I could guarantee 48 hours of teamwork to observe. How pleasantly surprised was I to find out at the end of the hackathon that individuals had dropped out, pivoted or merged. As I got wind of this mischief, my dissertation changed direction to focus on the moments of breakdown, resistance and play. A future direction in my research would track just the dropouts to challenge my findings thus far.

At the peak of the hackathon movement, several scholarly groups had attempted to survey the hackathon populations (Bernstein 2015; Major League Hacking 2015), which could benefit from a deeper analysis as well. Castells (1996, 372) reminds us of the widespread divide between self-programmable labor – those who have received an education that allows them to constantly redefine the necessary skills for a given task and better access the resources for learning additional skills – and the resulting exclusion of generic labor – the non-programmers. This dissertation should have given you a taste of what it is like to learn how to learn, but in the quest for the spectacular, it is easy to forget that my hackathon participants came from

prestigious universities and wealthy cities, therefore suggesting that the hackathon movement concentrates play labor at the hands of urban, American young elites. It would also be worthwhile to wait a number of years to see if this movement waxes and wanes, keeping in mind that while school is still in session, the majority of us cannot participate in hackathons because, despite being so magical, it is so much hard work.

References

Appadurai, Arjun. *The Social Life of Things: Commodities in Cultural Perspective*. Cambridge University Press, 1986.

Austin, John L. *How to Do Things with Words: The William James Lectures delivered at Harvard University in 1955*. Oxford: Clarendon Publishing, 1962.

Bakhtin, Mikhail. *Rabelais and His World*. Bloomington: Indiana University Press, 1941.

Bateson, Gregory
---------"About Games and Being Serious." In *Steps to an Ecology of Mind: Collected Essays in Anthropology, Psychiatry, Evolution, and Epistemology*. San Francisco: Chandler, 1972.

---------"A Theory of Play and Fantasy." In *Steps to an Ecology of Mind: Collected Essays in Anthropology, Psychiatry, Evolution, and Epistemology*. San Francisco: Chandler, 1972.

Beller, Jonathan
---------*The Cinematic Mode of Production: Attention Economy and the Society of the Spectacle*. Hanover: Dartmouth College Press, 2006.

---------"Informatic Labor in the Age of Computational Capital," *Lateral* 5 no. 1 (Spring 2016). http://csalateral.org/issue/5-1/informatic- labor-computational-capital-beller/.

Berg, Joyce, John Dickhaut and Kevin McCabe. "Trust, Reciprocity, and Social History." *Games and Economic Behavior* 10 no. 1 (Summer 1995): 122-142. doi: 10.1006/game.1995.1027.

Berry, Robert and Sebastian Cevey. "Hacking Journalism at the MIT Media Lab." *The Guardian*, July 18, 2014. https://www.theguardian.com/info/developer-blog/2014/jul/18/hacking-journalism-at-the-mit-media-lab.

Bernstein, Ethan S. "The Smart Way to Create a Transparent Workplace." *Wall Street Journal*, February 22, 2015. https://www.wsj.com/.

Boon, James A. *Verging on Extra-Vagance*. Princeton: Princeton University Press, 1999.

Brennan, Karen and Mitchel Resnick. "Imagining, Creating, Playing, Sharing, Reflecting: How Online Community Supports Young People as Designers of Interactive Media." In *Emerging Technologies for the Classroom*. Edited by Nancy Lavigne and Chrystalla Mouza, 253-268. New York: Springer, 2013.

Brunton, Finn and Gabriella Coleman. "Closer to the Metal." In *Media Technologies: Essays on Communication, Materiality, and Society*. Edited by Tarleton Gillepsie, Pablo J. Boczkowski and Kirsten A. Foot, 77-98. Cambridge: MIT Press, 2014.

Caillois, Roger. *Les Jeux et les Hommes*. Paris: Gallimard, 1958.

Caplan, Robyn and Danah Boyd. "Who controls the public sphere in an era of algorithms? Mediation, Automation and Power." Presentation at the Data Society Algorithms and Publics Workshop, New York, NY, February 2016.

Castells, Manuel. *The Network Society*. Malden: Blackwell, 1996.

Chun, Wendy Hui-Kyong. *Control and Freedom: Power and Paranoia in the Age of Fiber Optics*. Cambridge: MIT Press, 2006.

Crary, Jonathan. *24/7: Late Capitalism and the Ends of Sleep*. London: Verso Books, 2013.

Coleman, Gabriella E. "Anonymous: From the Lulz to Collective Action." *The New Everyday Media Commons Project*. Fall 2011. http://mediacommons.futureofthebook.org/.

---------"Ethnographic Approaches to Digital Media." *Annual Review of Anthropology* 39 (Fall 2010): 487- 505. doi: 10.1146/annurev.anthro.012809.104945.

--------"Hackers." *The Johns Hopkins Encyclopedia of Digital Textuality*. Edited by Marie-Laure Ryan, Lori Emerson and Benjamin Robertson, 245-248. Baltimore: John Hopkins University Press, 2014.

---------"Hacking In-Person: The Ritual Character of Conferences and the Distillation of a Life-World." *Anthropology Quarterly* 83 no. 1 (Winter 2010): 47-72.

Crogan, Patrick with Samuel Kinsley. "Paying Attention: Toward a Critique of the Attention Economy." *Culture Machine* 13 (2012): 1- 29. http://www.culturemachine.net.

Cubitt, Sean. "The Political Economy of Cosmopolis." In *Digital Labor: The Internet as Playground and Factory*. Edited by Trebor Scholtz, 58-68. New York: Routledge, 2012.

Currie, Morgan, Christopher Kelty, Luis Felipe Rosado Murillo. "Free Software trajectories: from organized publics to formal social enterprises?" *Journal of Peer Production* 3 (Summer 2013). Last accessed September 1, 2017, http://peerproduction.net.

Daly, Angela. "Disruption and the Law." *Journal of Peer Production* 6 (Winter 2015). http://www.peerproduction.net.

De Kosnik, Abigail. "Fandom as Free Labor." In *Digital Labor: The Internet as a Playground and a Factory*. Edited by Trebor Scholtz, 98-111. New York, NY: Routledge, 2012.

Dean, Jodi. "Whatever blogging." In *Digital Labor: The Internet as a Playground and a Factory*. Edited by Trebor Scholtz, 127-146. New York, NY: Routledge, 2012.

Debord, Guy. *La Société du Spectacle*. Paris: Folio, 2015.

Deleuze, Gilles. “Postscript on the Societies of Control.” *October* 59 (1992) :3-7.

D’Ignazio, Catherine, et al. “A Feminist HCI Approach to Designing Postpartum Technologies: ‘When I first saw a breast pump I was wondering if it was a joke.’” Paper Presented at the ACM 2016 Conference on Computer-Human Interaction, San Diego, CA May 2016.

Dorian, Nancy. *Investigating Variation: The Effects of Social Organization and Social Setting*. Oxford: Oxford University Press, 2010.

Durkheim, Emile. *The Elementary Forms of Religious Life*. Oxford: Oxford University Press, 2001.

Elias, Jennifer. “Why You Should Probably Host a Hackathon.” *Fast Company* May 1, 2014.

Fish, Adam et al. “Birds of the Internet: Towards a Field Guide to the Organization and Governance of Participation.” *Journal of Cultural Economy* 4 no. 2 (Spring 2011): 157-187. doi: 10.1080/17530350.2011. 563069.

Fisher, Tyler. “Subverting the Story Model. *Open News*, March 31, 2017. https://source.opennews.org/articles/subverting-story-mode.

Foucault, Michel. *Discipline & Punish: The Birth of the Prison,* 2nd ed. New York: Vintage Books, 1995.

Galloway, Alexander R. *Protocol: How Control Exists after Decentralization*. Cambridge, MIT Press, 2004.

Garfinkel, Harold. *Ethnomethodology's Program: Working Out Durkheim's Aphorism*. Lanham: Rowman & Littlefield Publishers, 2002.

Gee, James P. *What Video Games Have to Teach Us about Learning and Literacy*. London: Macmillan, 2014.

Geertz, Clifford. “Deep play: Notes on the Balinese cockfight.” In *The Interpretation Cultures: Selected Essays*, 80-98. New York: Basic Books, 1973.

Gershon, Iliana. “Neoliberal Agency.” Current Anthropology 52, no. 4 (Summer 2011): 537-555. doi: 10.1086/660866.

Girard, Monique and David Stark. “Heterarchies of value in Manhattan-based new media firms.” *Theory, Culture & Society* 20 no. 3 (Summer 2003): 77-105. doi: 10.1177/0263276430 203006.

Goffman, Erving. *The Presentation of the Self in Everyday Life*. New York: Random House, 1956.

Gumperz, John. "The Speech Community." In *Linguistic Anthropology: A Reader*. Edited by Alessandro Duranti, 43-52. Oxford: Blackwell, 2001.

Gumperz, John and Dell H. Hymes. *Directions in Sociolinguistics: The Ethnography of Communication*. New York: Holt, Rinehart and Winston, 1972.

Halliday, Michael A.K. *Language as a Social Semiotic*. London: Arnold, 1978.

Haywood, Douglas. "The Ethic of the Code: An Ethnography of a 'Humanitarian Hacking' Community." *Journal of Peer Production* 3 (Summer 2012). http://peerproduction.net.

Hoek, Lotte. *Cut-Pieces: Celluloid Obscenity and Popular Cinema in Bangladesh*. New York: Columbia University Press, 2014.

Horst, Heather A. and Daniel Miller. *Digital anthropology*. New York: Berg, 2012.

Huizinga, Johan. *Homo Ludens: A Study of the Play-Element in Culture*.

Hung, Aaron Chia-Yung
2011 *The Work of Play: Meaning-Making in Videogames (New Literacies and Digital Epistemologies)*. New York: Peter Lang, 2011.

Ingold, Tim and Jo Lee Vergunst. *Ways of Walking: Ethnography and Practices on Foot*. Burlington: Ashgate, 2008.

Irani, Lily. "Hackathons and the Making of Entrepreneurial Citizenship." *Science, Technology & Human Values* 40 no. 5 (Spring 2015): 799-824. doi: 10.1177/0162243915578486.

Ito, Mizuko et al. *Hanging Out, Messing Around, and Geeking Out: Kids Living and Learning with New Media*. Cambridge: MIT Press, 2009.

Johnson, Peter and Pamela Robinson. "Civic Hackathons: Innovation, Procurement, or Civic Engagement?" *Review of Policy Research* 31 no. 4 (Summer 2014): 349-357. doi: 10.1111/ropr.12074.

Johri, Al. "Domain Specific Newsbots - Live Automated Reporting Systems involving Two Way Natural Language Communication." Paper Presented at Stanford Computation + Journalism Conference, Stanford, October 2016.

Jones, Graham M., Beth Semel and Audrey Le. "'There Are No Rules: It's a Hackathon!': Negotiating Commitment in a Context of Volatile Sociality." *Journal of Linguistic Anthropology* 25 no. 3 (Winter 2015): 322–345. doi: 10.1111/jola.12104.

Kafai, Yasmin B. and Quinn Burke. *Connected Code: Why Children Need to Learn Programming*. Cambridge: MIT Press, 2014.

Kelty, Christopher. "Geeks, Social Imaginaries, and Recursive Publics." *Cultural Anthropology*

20 no. 2 (Spring 2005): 185-214. doi: 10.1525/can.2005.20.2.185.

--------*Two Bits: The Cultural Significance of Free Software*. Duke University Press, 2008.

Kilburn, Lilia. "Hacking Institutional Corruption." MIT Center for Civic Media (blog). March 30, 2015. https://civic.mit.edu/blog/liliamaud/hacking-institutional-corruption.

Kliff, Sarah. "Everything You Need to Know about Obamacare." Card Stack, Vox, January 5, 2017. https://www.vox.com/cards/obamacare.

Koyama, Jill P. and Hervé Varenne. "Assembling and Dissembling: Policy as Productive Play." *Educational Researcher* 41 no. 5 (Summer 2012): 157-162. doi: 10.3102/0013189X12442799.

Lave, Jean, and Etienne Wenger. *Situated Learning: Legitimate Peripheral Participation*. New York: Cambridge University Press, 1991.

Larkin, Brian. "Degraded Images, Distorted Sounds: Nigerian Video and the Infrastructure of Piracy." Public Culture 16 no. 2 (Spring 2004): 289-314. doi:10.1215/08992363-16-2-289.

Latour, Bruno. *Reassembling the Social: An Introduction to Actor-Network Theory*. Oxford: University Press, 2005.

Lazzarato, Maurizio. "Immaterial Labour." In *Radical Thought in Italy: A Potential Politics*. Edited by Paolo Virno and Michael Hardt, 133-147. Minneapolis: University of Minnesota Press, 1996.

Major League Hacking. "Frequently Asked Questions." Last Accessed on September 2, 2017, https://mlh.io/faq.

Marx, Karl. *Grundrisse: Foundations of A Critique of the Political Economy*. Marxist Archives, 1973. https://www.marxists.org/archive/marx/works/1857/grundrisse/.

Mauss, Marcel. *Essai sur le Don*. Paris: Presses Universitaires de France, 2007.

McDermott, Ray and Hervé Varenne. "Culture as disability." Anthropology & Education Quarterly 26 no. 3 (Fall 1995): 324-348. doi: 10.1525/aeq.1995.26.3.05x0936z.

Merleau-Ponty, Maurice. *The Prose of the World*. Northwestern University Press, 1973.

Michael and Tom McEney. "Language-in-use and literary fieldwork." *Representations* 137 no. 1 (Winter 2017): 1-22. http://www.representations.org/language-in-use-and-literary-fieldwork/

Microsoft News Centre UK. "Hospital Uses Wearable Device to Create New Healthcare Reality

for People with Epilepsy." Microsoft (blog). February 7, 2017. https://www.microsoft.com/empowering-countries/en-us/good-health-and-well-being/hospital-uses-wearable-device-to-create-new-healthcare-reality-for-people-with-epilepsy/.

Miller, Stephen. "Ends, Means, and Galumphing: Some Leitmotifs of Play." *American Anthropologist* 75 no. 1 (Winter 1973): 87-98. doi: 10.1525/aa.1973.75.1.02a00050.

Mitchell, Amy, Jesse Holcomb and Rachel Weisel. *The State of the News Media 2016*. Washington D.C.: Pew Research Center, 2016.

Negri, Antonio and Michael Hardt. *Empire*. Cambridge: Harvard University Press, 2000.

---------*Multitude: War and Democracy in the Age of Empire*. New York: Penguin, 2004.

Nguyen, Lilly U. "Infrastructural Action in Vietnam: Inverting the Techno-politics of Hacking in the Global South. *New Media & Society* 18 no. 4 (Winter 2016): 637-652. doi: 10.1177/1461444816629475.

The New York Times. "Journalism That Stands Apart." Report published by the 2020 Group. January 17, 2017. https://www.nytimes.com/projects/2020-report/?mcubz=0.

Ortner, Sherry B. *Making Gender: The Politics and Erotics of Culture*. Boston: Beacon Press, 1997.

Overly, Steven. "Universities Are Venturing into New Territory: Funding Start-up Businesses." *The Washington Post*, February 14, 2015. https://www.washingtonpost.com.

Peterson, Tim. "Vox.com Aims to Bring Context to News With 'Card Stacks: Wikipedia-Like Entries On Hot Topics Maintained by Experts." Vox, April 6, 2014. http://adage.com/article/digital/vox-aims-bring-context-news-card-stacks/292514/

Piaget, Jean. Play, Dreams and Imitation in Childhood. New York: Norton Library, 1945.

Raatikainen, Mikko, et al. "Industrial experiences of organizing a hackathon to assess a device-centric cloud ecosystem." Paper presented at the 37th Annual Computer Software and Applications Conference, Kyoto, July 22-26, 2013.

Richard, Gabriella T., et al. "StitchFest: Diversifying a College Hackathon to Broaden Participation and Perceptions in Computing." SIGCSE '15 Proceedings of the 46th ACM Technical Symposium on Computer Science Education, Kansas City, MO, March 2015. http://dl.acm.org/citation.cfm?id=2677310

Roberts, Edward B., Fiona Murray, and J. Daniel Kim. *Entrepreneurship and Innovation at MIT Continuing Global Growth and Impact*, a report published in 2015 by the MIT Martin Trust Center for Entrepreneurship.

Ross, Andrew. "In Search of the Lost Paycheck." In *Digital Labor: The Internet as Playground and Factory*. Edited by Trebor Scholtz, 13-32. New York: Palmgrave. 2012.

Salen, Katie et al. *The Ecology of Games: Connecting Youth, Games, and Learning*. CambridgeL MIT Press, 2008.

Santo, Rafi. "Hacker Literacies: User-Generated Resistance and Reconfiguration of Networked Publics." In *Critical Digital Literacies as Social Praxis: Intersections and Challenges*. Edited by Colin Lankshear and Michele Knobel, 197-217. New York: Peter Lang, 2012.

Scholz, Trebor. "Introduction: Why Does Digital Labor Matter Now?" In *Digital labor: The Internet as Playground and Factory*. Edited by Trebor Scholtz, 1-10. New York: Routledge, 2012.

Schrage, Michael and Tom Peters. *Serious Play: How the World's Best Companies Simulate to Innovate*. Cambridge: Harvard Business School Press, 1999.

Silverstein, Michael. "The Limits of Awareness," in Linguistic Anthropology: A Reader. Edited by Alessandro Duranti, 382–401. Malden: Blackwell, 2001.

Stoltfuz, Arlin et al. "Community and Code: Nine Lessons from Nine NESCent Hackathons." Paper Presented at the 20th ACM Conference on Computer-Supported Collaborative Work, Portland, OR, February 25-March 1, 2017.

Suchman, Lucille Alice. *Human-machine reconfigurations: plans and situated actions*. New York: Cambridge University Press, 2007.

Sullivan, Mark. "Fitness Tracker Sales Will Triple by 2018, Then Smartwatches Take Over." *Venture Beat*, November 25, 2014. https://venturebeat.com/

Sutton-Smith, Brian. *The Ambiguity of Play*. Cambridge: Harvard University Press, 1997.

Taylor, Charles. *Modern Social Imaginaries*. Durham: Duke University Press, 2003.

Terranova, Tiziana. "Capture All Work." Filmed January 31, 2015 in Berlin. *Art & Education*, October 26, 2015. Video, 29:27. http://www.artand education.net/videos/tiziana-terranova-capture-all-work/

-------- "Free labor: Producing culture for the digital economy." *Social Text* 18 no. 2 (Summer 2000): 33-58.

-------- *Network Culture*. Pluto Press, 2004.

Tinoco, Christian. "A Mega-City on the Verge of Collapse." Card Stack, FOLD, December 5, 2016. https://fold.cm/read/christn7/a-mega-city-on-the-verge-of-collapse-smwePuqd.

Tosay, Paul. "Bateson's Levels Of Learning: a Framework For Transformative Learning?" Paper presented at Universities' Forum for Human Resource Development Conference, Tilburg,

May 2006. https://epubs.surrey.ac.uk/1198/1/fulltext.pdf.

Tufecki, Zeynep, with Lucy Suchman. Interview by Major Garrett. Charlie Rose, July 6, 2016.

Turner, Victor. *From Ritual to Theater*. Ann Arbor: Performing Arts Journal Publications, 1982.

Varenne, Herve and Ray McDermott. "The Production of Difference in Interaction: On Culturing Conversation through Play." In *Theoretical Approaches to Dialogue Analysis*. Edited by Lawrence N. Berlin, 177-197. Tübingen: Max Niemeyer Verlag, 2007.

Virno, Paolo. "Labour and Language." In *Lexicon of Postfordism*. Edited by Adelino Fadini and Ubaldo Zanini. Milan: Feltrinelli, 2001. http://www.generation-online.org.

"Virtuosity and Revolution." In *Radical Thought in Italy: A Potential Politics*. Edited by Paolo Virno and Michael Hardt, 189-212. Minneapolis: University of Minnesota Press, 1996.

Vygotsky, Lev. Play and its role in the Mental development of the Child. Marxist Archives, 1933. https://www.marxists.org/archive/vygotsky/works/1933/play.htm

Wark, Makenzie. A Hacker's Manifesto. Cambridge: Harvard University Press, 2004.

World Bank "IT-Based Innovation in Rural and Urban Sanitation Hackathon. "A report published in 2014 by The World Bank.

LIBOR Alt + Repair

Judge 1: Scored 5-3-4-1-3. (Bottom of the page) "Not sure what was built? Data viz available to use?"

Judge 2: Scored 5-4-4-3-4.

APPLICABILITY – rated 5, Yes offers alternative rate.

NECESSITY/IMPACT – rated 4, Alternative to London Inter-bank offer rate based on market data —> six alternatives. Pulls data daily —> demand-driven solution.

PRACTICALITY – rated 4, Live website, explanation.

OPENNESS/REPLICATION – rated 3, Not sure.

ACCESSIBILITY – rated 4, Yes it's clearly explained/Laid out on website.

Judge 3: Scored 3-2-1-2-2.

APPLICABILITY – rated 3, View current day benchmark interest rates. Address- yes. Mitigate- unlikely.

NECESSITY/IMPACT – rated 2, Libor subject to collusion. - Rates consistently overstated + manipulation. - Create alternative benchmarks.

PRACTICALITY – rated 1, (circled "in the near term") OPENNESS - 2, LIBOR-specific.

ACCESSIBILITY - 2, "users" would have to demand/request this new rate be used. What is motivation/stick for lenders?"

Judge 5: Scored 3-3-4-3-4. No comments.

Judge 6: Scored 2-2-5-3-3. (On top of page) LIBOR Transparency. Provide a benchmark rate for the LIBOR.

APPLICABILITY – rated 2, Not sure that going against LIBOR is an institutionalized corruption problem - just about a broken market.

NECESSITY/IMPACT – rated 2, Theory of change very unclear, not clear why transparency would make a difference.

PRACTICALITY – rated 5, Team built the demo, + log of how it works is pretty clear.

ACCESSIBILITY – rated 3, Not sure who would use this….

Judge 7: Scored 5-4-5-3-3. PRACTICALITY – rated 5, 6 alt. benchmark rates

WeCott

Judge 1: "Very nice. Working. Simple idea. Corruption. Builds community." Two stars.

Judge 2: Scored 4-4-5-5-4.

APPLICABILITY - 4, It provides a tool for one solution.

NECESSITY - 4, Problems: Corporate corruption + influence on people. Use a social network to gain support (pledge boycotting amounts).

PRACTICALITY - 5, Live website —> Great design.

OPENNESS - 5, Yes could be used for politics.

ACCESSIBILITY - 4, Yes very easy to use, the problem will be attracting users.

Judge 3: Scored 4-4-5-5-5. Circled team name "Enabling the public to stand up to corp corruption" NECESSITY - 4, Corp. Practices vs. Public good. Bullet list: - unfair prices - poor worker treatment - unethical practices. Supply chain issues. Bullet list: - treatment of unions - slave labor.
PRACTICALITY - 5, Boycotting barriers, bullet: awareness.
ACCESSIBILITY - 5, "Comcast is evil" Boycott public supermarket. Bullet list: - mobile app - barcode - photo showing. Next to circled score: -Elegant simplicity - Resources diverted dashboard
Judge 4: Scored 4-2-4-4-4
Judge 5: Scored 4-4-4-4-4.
PRACTICALITY - 4, I worry people coming up with boycotts that should not be started.
Judge 6: Scored 4-4-5-5-4. Create + social media platform for boycotts. Target - corporate corruption. ETHICAL CONSUMERISM.
APPLICABILITY - 4, Good project - but most corporate malfeasance is not really institutionalized corruption. NECESSITY - 4, Problem is clear.
PRACTICALITY - 5, Demo is up + live. Very cool. Also, very well thought-out. They got really far on this demo. Nice job.
OPENNESS- 5, Really nice adaptation of change, organization of idea etc.
ACCESSIBILITY- 4, Need lots of consumers, but probably will be costly. Starred. Boxed "winner"
Judge 7: Scored 4-5-5-5-5.
PRACTICALITY - 5, Live now, see $ diverted

CampaignCon
Judge 1: "Confused about the actual issue. Data viz is interesting. Tool? DV for selves?"
Judge 2: Scored 3-4-5-4-4 "System workflow pulls FEC data, parse data, look for suspicious data/changes, publish changes + interactives.
APPLICABILITY - 3, It loosely does. How would this influence?
NECESSITY - 4, Problem: How campaign contributions change over time. Some campaign data is deleted —> $7.5m between 2008-12 calls out the changes/discrepancies.
PRACTICALITY - 5, Yes.
OPENNESS/REPLICATION - 4, Could be used for other data points.
ACCESSIBILITY - 4, Potentially.
Judge 3: Scored 4-4-3-4-3. Congress, Contributions, Con jobs.
APPLICABILITY - 4, Demonstrate growth of contributions in size + number.
NECESSITY - 4, Good data viz.
PRACTICALITY - 3, ? (Circled financially)
Judge 4: Scored 4-4-4-4-2
Judge 5: Scored 5-5-5-5-5.
APPLICABILITY - 5, Loved it! Just the nature of having this new system people receiving

campaign contributions are going to think carefully about their actions.

Judge 6: Scored 5-3-4-5-4. Idea: FEC dat are deleted. Data integrity [WAIVER] —> Campaign finance visualization tool (1) retrieve weekly updates (2) look for changes + interactive visualizations.

PRACTICALITY - 4, seems like they already uploaded. Would like to hear about dark $

Judge 7: Scored 5-5-4-2-3.

NECESSITY - 5, donations deleted from FEC data.

PRACTICALITY - 4, download FEC filings each week. Figure out what the changes are —> What's suspicious —> Publish & contact interested parties