utopian hacks

Not all engineers create equally. GÖTZ BACHMANN takes us inside the labs of “radical engineers” and the starkly different futures they imagine for us.

ABOVE: Illustration (draft) by David Hellman, imagining jointly with Bret Victor's group "Dynamic Land", dynamic spatial media's next iteration in 2017.

IN A LAB IN OAKLAND a group of engineers are building a "new kind of computer." It is here, in this lab in Silicon Valley (or in close proximity to Silicon Valley, depending where you draw its boundaries), that I base my ongoing ethnography. The group, clustered around an engineer named Bret Victor, is part of YC Research's Human Advancement Research Community (HARC), an industry-financed research lab devoted to open and foundational research. "Hacking" is for the members of this group, just as it is for many other engineers, at best a word for tentative work (as in: "This is just a hack") or for using technologies for other purposes than those originally intended for them. It can also be a derogatory term for not thinking through the consequences of the accumulation of amateurish, low-quality tech development. Thus: when the engineers I research describe their work, "hacking" would not be one of the key terms they would choose. However, I want to make the case that some of their work practices share similarities with hacking, albeit in a different realm. This article asks: How do engineers hack imaginaries of what technologies are and can be?

I argue this claim by analyzing these engineers as part of a tradition which I call, for lack of a better term, "radical engineering." Radical engineers fundamentally challenge existing notions of (here, digital media) technologies: their basic features, purposes, and possible futures. Their radicality is not to be confused with political radicality, or the radicality of "disruption", or the radicality of some of engineering's outcomes. Theirs is a radicality that puts them outside of assumptions in the wider engineering field of what is obvious, self-evident, time-tested or desirable. Their positions are so heterodox that they often stop calling themselves "engineers." But no other word can take its place. They might experiment with words like "artist" or "designer in the Horst Rittel way," but neither stabilizes and both are prone to cause misunderstanding. After all, the people at stake here have their education in disciplines like electrical engineering, mechanical engineering, computer science or mathematics, and their work often comes with the need to tackle highly complex technical problems.

Bret Victor's group tries to build a new computational medium. To get there is less a question of a sudden eureka, but more a permanent and stubborn process of pushing beyond what is thinkable now. The lab takes existing technologies such as projectors, cameras, lasers, whiteboards, computers, and Go stones, and recombines them with new or historic ideas about programming paradigms, system design and information design, as well as a range of assumptions and visions about cognition, communication, sociality, politics and media. The group is constructing a series of operating systems for a spatial dynamic medium, each building on the experiences of building the last one, and each taking roughly two years to build. The current OS is named "Realtalk" and its predecessor was called "Hypercard in the World" (both names pay respect to historical, heterodox programming

environments: Smalltalk in the 1970s and Hypercard in the 1980s). While the group develops such operating systems, it engages in a process of writing and rewriting code, as well as manifestos, lots of talking, even more moments of collective silence, of iterating and tweaking mantras, of digesting films and books ,as well as huge amounts of technical papers, and building dozens—indeed hundreds—of hardware and software prototypes.

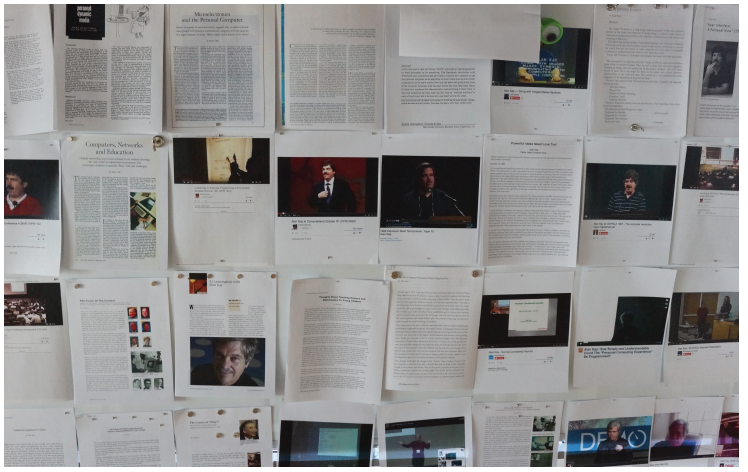
The lab is filled with prototypes, and new ones are added by the week. In one month, a visitor is able to point a laser at a book in the library, and a projector beams the inside of that book on the wall next to her. A few weeks later you will see people jumping around on the floor, playing "laser socks": a game where people try to laser each other's white socks. Months later, a desk becomes a pinball machine made out of light from a projector, and cat videos follow around every rectangle drawn on a piece of paper. Currently, the group experiments with "little languages" in the spatial medium: domain specific programming languages based on paper, pen and scissors, Go stones, or wires, all equipped with dynamic properties, thus having capabilities to directly steer computation or visualize complexity. The point of all such prototypes is not technical sophistication of the glitzy kind. In fact, it is the opposite. The prototypes aim for simplicity and reduction—as a rule of thumb, you can assume that the fewer lines of code involved, and the simpler these lines are, the more the prototype is deemed successful.

In all their playfulness, these prototypes remain "working artefacts" (Suchman et al 2002, 175), forming "traps" for potentialities with "illusions of self-movement" (Jiménez 2014, 391). In the research group of Bret Victor, the work of prototypes is to catch and demonstrate potential properties of a new, spatial, dynamic medium. As one of its desired properties is simplicity, those prototypes that show this property tend to be selected as successful. Furthermore, in the last four years the group has built two operating systems, and aims to keep up this tempo. Each experimental operating system is a prototype, too, albeit a bigger and more complex one. But it is also a purpose-built environment for prototyping applications. And the operating system is based on lessons from past prototyping, including prototyping of both applications and operating systems. These lessons consist of the exploration of desirable, new properties of applications and operating systems. If successful, a new operating system allows building new prototype applications with the desired properties. At the same time, these experiences might point to further desirable properties. This process is then iterated. The overall goal is to create a rupture of a fundamental kind, a jump in technology equivalent to the jump in the 1960s and early 1970s when the quadruple introduction of the microprocessor, the personal computer, the graphical user interface, and the internet revolutionized what computing could be by turning the computer into a medium. Turning computing into media was already in the 1960s and 1970s meant to work with technology against technology: by using new computational capabilities, a medium was carved out that complies less with perceptions at the time of what computing "is," and more with what a medium that forms a dynamic version of paper could look like. This form of working *with* computing *against* computing is now radicalized in the work of Bret Victor's research group.

The patron saint for this enterprise, both in spirit and as a real person, is Alan Kay, one of the most famous radical engineers and a key contributor to those ruptures in computing in the 1960s and 1970s that Bret Victor's group tries to match today. So let's zoom in on Kay. He started his work in the 1960s at the newly founded Computer Science Department at the University of Utah, writing what surely was one of the boldest doctoral dissertations ever written, a wild technological dream of a new form of computing. A reference to another radical engineer's cry of despair—"I wish these calculations were executed by steam" (attributed to Charles Babbage and quoted in Kay 1969, III)—stands at its beginning, and after 250 pages of thinking through a "reactive engine," it culminates in a "handbook" for an imaginary "Flex Machine": a first iteration of a set of ideas that culminated a few years later in Kay's vision for a "DynaBook" (1972). While still working on this thesis, Kay became one of the Young Turks in the research community funded by the Pentagon's Advanced Research Project Agency's (ARPA) Information Processing Techniques Office (IPTO), which was at that time making its first steps towards building the ARPANET. In the early 1970s, after a quick stint as a postdoc with John McCarthy at Stanford, Kay joined Bob Taylor's new Xerox PARC research lab, where engineering legends such as Lampson, Thacker, Metcalfe, and many others, were building the ALTO system, which was the first system of connected standalone machines with advanced graphic abilities.

Once the first iterations of the ALTO/Ethernet system—and it is essential to understand the latter as a system and not as standalone computers—were up and running, they provided Kay with a formidable playground. Kay went back to some of his work in the 1960s, when he had analyzed SIMULA (an obscure Norwegian programming language), and developed this, with Dan Ingalls and Adele Goldberg, among others, into a hybrid between a programming language, an operating system, and a kid's toy called Smalltalk. The first iterations of Smalltalk were experiments in object orientation that aimed to model all programming from scratch after a distributed system of message passing (Kay 1993): later versions gave up on this, and after an initial phase of success Smalltalk eventually lost the battle over the dominant form of object orientation to the likes of C++ and Java. But in the mid 1970s the ALTO/Ethernet/Smalltalk system became a hotbed for an explosion of ideas about the graphical user interface (GUI) as well as dozens of now common applications. The work of Kay and his "Learning Research Group" can thus be seen as both a lost holy grail of computing before it was spoiled by a model of computing as capitalism cast in hard- and software, but also as one of the crucial genealogical hubs for its later emergence. And it is this double meaning that makes this work so unique and interesting to this day.

Alan Kay's contributions to the history of computing are results of radical hacks of the computational paradigms and imaginaries of his time. Kay took heterodox programming techniques like the one pioneered by SIMULA, new visualization techniques like the ones developed by the Sutherland brothers, McCarthy cravings for "private computing" (1962:225) and Wes Clark's lonely machines, the experiments in augmentation by Doug Engelbart's group, and new ideas about distributed networks, to name a few. Such techniques were not common sense in the emerging professions of software engineering and programming, but had started to circulate in the elite engineering circles where Kay worked. Kay combined them with ideas about pedagogy,



ABOVE LEFT: A whiteboard in the lab of Bret Victor's group filled with papers by Alan Kay.
ABOVE RIGHT: A detail in the HARC Lab: Above, Alan Kay, in white jeans. Below: Engelbart's 1962 paper, glued on a wall in San Francisco's Mission district by Bret Victor.

psychology, and mathematics by Maria Montessori, Seymour Papert, and Jerome Bruner, and added further zest in form of the sassy media theoretical speculations of Marshall McLuhan. Kay was also very early in understanding the implications of what Carver Mead called "Moore's Law," an exponential line of ever smaller, faster, and cheaper forms of computing kicked off by the mass-produced integrated circuit, and now leading to the positive feedback of technical development and the creation of new markets. So Kay took all of these ideas, desires, technologies, and opportunities, and recombined them. The results were crucial contributions to a new and emerging sociotechnical imaginary, in many ways representing the computer as a digital medium, which we now have today. Kay's work can thus be seen as a benchmark in radical engineering, as such enabling us to critique the stalemate and possible decline in quality of most currently available imaginaries about technologies.

But is it really that easy? Is radical engineering simply the result of a bit of remixing? Obviously it is a much more complicated process. One of the most convincing descriptions of this process stems from another legendary radical engineer, the aforementioned Doug Engelbart. In 1962, a few years before Alan Kay started his career, Engelbart set the program for his own U.S. Air Force-funded research group at the Stanford Research Institute (Bardini 2000:1-32), aiming for nothing less than to re-engineer the "HLAM-T," the "Human using Language, Artifacts, Methodology, in which he is Trained" (Engelbart 1962:9). This HLAM-T was always a cyborg, and as such it can be engaged in a continuous process of "augmenting human intellect." According to Engelbart, the latter can be achieved through the process of "bootstrapping." This is a term that can mean many things in the Silicon Valley, from initiating systems to kicking off startups, but in the context of Engelbart's work, bootstrapping is the "...interesting (recursive) assignment of developing tools and techniques to make it more effective at carrying out its assignment. Its tangible product is a developing augmentation system to provide increased capability for developing and studying augmentation systems" (Engelbart and English 2003:234). Just as Moore's so-called law, this is a dream of exponential progress emerging out of nonlinear, self-enforcing feedback. How much more Californian can you be?

For Engelbart and English's description to be more than just a cybernetic pipedream, we need to remind ourselves that they were not only speaking about technical artifacts. Bootstrapping is a larger process in which "tools and techniques" are developing with social structures and local knowledge over longer periods of time. The processes are recursive, much like the "recursive publics" that Chris Kelty (2008:30) describes for the free software development community: in both cases developers create sociotechnical infrastructures with which they can communicate and cooperate, which then spread to other parts of life. Kelty shows how such recursive effects are not simply the magical result of self-enforcing positive feedback. Recursive processes are based on politics. And resources. And qualified personnel. And care. And steering. In short, they need to be continually produced.

As such, bootstrapping can assume different scopes and directions. Bret Victor and his research group's form of bootstrapping resembles a multi-layered onion. The kind of people who should be part of it, and at what moments, can lead to intense internal discussion. Once the group launches "Dynamic Land" (see image), it will reach its next stage (to be described in a future paper). Meanwhile, bootstrapping has already taken many forms. Prototypes relate to the process of bootstrapping as pointers, feelers, searchers, riffs, scaffolds, operating systems, jams, representations, imaginary test cases, demos and so on. The interplay of prototype operating systems and prototype applications drives the process forward. Forms of working and cooperation inside the group are evolving, too. There is, indeed, a bestiary of prototyping techniques contained in the larger process of bootstrapping. Together, inside the lab, they produce a feeling of sitting inside a brain. The lab as a whole—its walls, desks, whiteboards, roofs, machines, and the people inhabiting it—functions as a first demo for an alternative medium.

Building the iterations of the series of operating systems can require substantial engineering tasks in the more classical sense; such as, for example, programming a kernel in C, or a process host in Haskell. But the overall endeavor is decidedly not driven by technology. In the spatial medium to come, computing is

supposed to be reduced. Computing is to take the role of an infrastructure: much as books need light, but are not modeled after the light's logic, the medium might draw, where necessary, on the computing possibilities provided by the OS in the background, but it should not be driven by them. Instead, the dynamic spatial medium should be driven by properties of the medium itself, and as such, it should drive technology. The medium's properties are yet to be explored by the very process of bootstrapping it. In the parlance of the group, both the medium and the ways in which they produce this medium, are "from the future." That future is not given, but depends on the medium the group is imagining. It thus depends on the properties of the medium that the group is exploring, selecting, and practicing. On the one hand, technology enables a new medium, which is imagined as shaping the future, on the other hand the future is imagined as shaping the new medium, which then should drive technology.

While most of the group's work consists of building devices, speculative thought is part of their work as well. The latter enables the engineers to understand what the prototyping work unveils. It also gives the lab's work direction, motivates its enterprise, and is part of acquiring funding. The overall process has by now led to a set of interconnected and evolving ideas and goals: One cluster looks, for example, for new ways of representing and understanding complex systems. A second cluster aims for more access to knowledge by undoing contemporary media's restrictions (such as the restriction of the screen, which produces, with its peek-a-boo access to complexity, impenetrable forms of knowledge such as the trillions of lines of code, written on screens and then stared at on screens). A third cluster explores new forms of representing time, and a fourth one more effective inclusion of physical properties into the spatial media system. All these clusters would lead, so the goal and the assumption, to more seamless travels up and down the "ladder of abstraction" (Victor 2011.) As if to echo Nietzsche's, McLuhan's, or Kittler's media theoretical musings with engineering solutions, a larger goal is to make new thoughts possible, which have until now remained "unthinkable" due to contemporary media's inadequacies. Enhanced forms of embodied cognition, and better ways of cooperative generation of ideas could cure the loneliness and pain that are often part of deep thought. And all of it together might, to quote an internal email, "prevent the world from taking itself apart."

One way to understand what's going on here is to frame all this as an alternative form of "hacking." When you "hack," you might be said to be hacking apart or hacking together. Hacking apart could then be seen as the practices evolving out of the refusal to accept former acts of black boxing. Transferred to radical engineering, hacking apart would translate into not accepting the black boxes of present technological paradigms such as screen-based computers, or ready-made futures such as, say, "Smart Cities, Smart Homes" or the "Internet of Things." Instead you would open such black boxes and dissect them: assumptions about what is deemed as technologically successful and about technological advances to come, matched by certain versions of social order, and often glued together with an unhealthy dose of business opportunity porn. The black boxes will most likely also contain ideas about the roles of the different types of engineers, programmers, designers, managers, and so on. If you take all this apart, you might look at the elements, throw away a lot of them, twist others, add stuff from elsewhere, and grow some on your own. You will look into different, often historical, technological paradigms, other ideas about what will become technologically possible (and when), different ideas of social order, the good life and problems that need addressing, other books to be read, alternative uses of the forces of media, and different ideas about the kind of people and the nature of their professions or non-professions, who should take charge of all this. If you are lucky, you have the conditions and abilities to work all this through in a process also known as bootstrapping, where you go through many iterations of hacking apart and hacking together, all the while creating fundamentally different ideas about what technologies should do, and could do, matched by a succession of devices and practices that help shape these ideas, and "demo" to yourself and others that some utopias might not be out of reach. This is what radical engineers do.

To prevent misunderstanding: neither I, nor the engineers I research, think that the actual future can be hacked together singlehandedly by a bunch of engineers in Palo Alto or Oakland. But I do think that radical engineers such as Engelbart's, Kay's, or maybe Victor's research groups, in their specific, highly privileged positions, add something crucial to the complex assemblage of forces that move us in the direction of futures. My ongoing fieldwork makes me curious about what is produced here, and many people who visit the lab agree that the first "arrivals" are stunning and mind boggling indeed. If we believe the group's self-perception, their technologies are, just like hacks, tentative interim solutions for something bigger that might arrive one day. The radical engineers would also be the first to state that the same interim solutions, if stopped in their development and reified too early, are potential sources of hacks in the derogatory sense. The latter is, according to their stories, exactly what happened when, 40 years ago, the prototypes left the labs too soon, and entered the world of Apple, IBM, and Microsoft, producing the accumulation of bad decisions that led to a world where people stare at smartphones.

Within such stories, radical engineers might employ a retrospective "could have been," a "Möglichkeitssinn" (sense of possibility, Musil 1930/1990, 14-18) in hindsight, mixed with traces of distinction against "normal" engineers. While they make considerable efforts to evade techno-solutionist fantasies, they don't abandon engineering's approach of addressing problems by building things. Even though they distance themselves from Silicon Valley's entrepreneurial cultures, their isolation against the "Californian ideology" (Barbrook 2007; Barbrook and Cameron 1995) might not always be 100% tight. Indeed, they might

BELOW: Alan Kay in a Japanese manga by Mari Yamazaki.

provide the Silicon Valley mainstream with the fix of heterodoxy it so desperately needs. Yet the same radical engineers are potential allies to those, who aim to hack apart the libertarian, totalitarian and toothless imaginaries that Silicon Valley so often provides us with, be it the "Internet of Shit" or the "crapularity" (Cramer 2016). The conceptual poverty of most of Silicon Valley's currently available futures surely can become visible from the perspectives of critical theory, from viewpoints of social movements, or through political economy's analysis. But Silicon Valley's timidity in thinking, which is only thinly veiled by the devastation it causes, also becomes apparent, if we compare it to radical engineering's utopias. ■

GÖTZ BACHMANN *is based at Leuphana University, Germany and is currently a Visiting Fellow at Stanford. He is an ethnographer, with former fieldwork among warehouse workers, saleswoman, and cashiers in Germany, and among Japan's Nico Chuu. He also authors the German children's comic series* KNAX.

BIBLIOGRAPHY

Barbrook, Richard. 2007. *Imaginary Futures: From Thinking Machines to the Global Village*. London, UK: Pluto.

Barbrook, Richard, and David Cameron. 1995. "The Californian Ideology." Mute 1(3) (republished in *Proud to be Flesh*, edited by Josephine Berry Slater and Pauline van Mourik Broekman, pp.27–34. London, UK: Mute Publishing)

Bardini, Thierry. 2000. *Bootstrapping. Douglas Engelbart, Co-evolution and the Origin of Personal Computing*. Stanford, CA: Stanford University Press.

Cramer, Florian. 2016. "Crapularity Hermeneutics." http://cramer.pleintekst.nl/essays/crapularity_hermeneutics/

Engelbart, Doug. 1962. *Augmenting Human Intellect: A Conceptual Framework. Summary Report*. AFO SR 3223. Stanford, CA: Stanford Research Institute.

Engelbart, Doug, and William English. 2003. "A Research Center for Augmenting Human Intellect." In *The New Media Reader*, edited by Noah Wardrip-Fruin, pp. 231–246. Cambridge, MA: MIT Press.

Jiménez, Alberto Corsín. 2014. "Introduction – The Prototype: More than many and less than one." In *Journal of Cultural Economy* 7(4):381–398

Kay, Alan C. 1969. "The Reactive Engine." PhD dissertation, The University of Utah, Salt Lake City.

———. 1972. "A Personal Computer for Children of all Ages." In *Proceedings of the ACM National Conference, Boston* (typed manuscript, no page numbers)

———. 1993. "The Early History of Smalltalk." *SIGPLAN Notices* 28(3):69–95.

Kelty, Chris. 2008. *Two Bits: The Cultural Significance of Free Software*. Durham, NC: Duke University Press.

McCarthy, John. 1962. "Time-Sharing Computer Systems." In *Management and the Computer of the Future*, edited by Martin Greenberger, pp. 221–236. Cambridge, MA: MIT Press.

Musil, Robert. 1930. *Der Mann ohne Eigenschaften* (The Man without Qualities.) Vol. 1. Berlin, Germany: Rowohlt.

Suchman, Lucy, Randall Trigg, and Jeanette Blomberg. 2002. "Working artefacts: ethnomethods of the prototype." In *British Journal of Sociology* 53(2):163–179.

Victor, Bret. 2011. "Up and Down the Ladder of Abstraction. A Systematic Approach to Interactive Visualisation" http://worrydream.com/LadderOfAbstraction/ accessed 8.2.17.