Beyond the Cruelty of Software

Galo Canizares

Barely a month into online teaching I noticed a new type of debate occurring during university faculty meetings. Suddenly, it seemed there were numerous ongoing arguments over which software would be best suited for distance learning. Should we use Zoom, Microsoft Teams, Google Meet, or the odd sounding BlueJeans? What's better, Slack or Blackboard? How does the Family Educational Rights and Privacy Act (FERPA) and security factor into our software preferences? It wasn't a simple argument over specific computer applications; individuals had to convince other faculty that their preferred tool was the best, the most powerful, the most productive, or the most efficient. Absent from these judgements on software, however, was any mention of compassion, community, pleasure, or delight. No one associated software with anything meant to spark joy. There was only productivity and efficiency.

While quarantine brought to the surface many broad systemic injustices and inequalities, it also revealed some smaller oppressive technological structures that had remained up until then largely invisible. Not only did online teaching call attention to a massive gap in internet access and digital surveillance systems reveal their nefarious role in policing, the software we engage with on a daily basis began to appear increasingly cruel in an already cruel world. Zoom's lack of security led directly to countless hateful *zoombombings*; Twitter's image algorithm was proven to actively suppress non-white skin tones; and despite initially pausing their Creative Cloud subscription fees, Adobe continued to hold their users hostage and dependent on their proprietary tools. Perhaps most unnoticed of all was the emergence of spyware-esque remote proctoring software, a somewhat organic by-product of institutionalized examination protocols sold to universities and testing companies to monitor at-home exams. Far from the dark web's version of villainy, these episodes constitute a kind of quotidian cruelty, an almost boring malevolence that most of us ignore as we attempt to remain productive. Our ability to look past these phenomena or perhaps dismiss them as simple glitches or errors is a testament to the normalization of software's cruelty. As Ruha Benjamin reminds us in *Race After Technology*, a software glitch is a "slippery place...between fleeting and durable, micro-interactions and macro-structures, individual hate and institutional indifference."[1] In other words, glitches and errors in systems reveal at best ignorance or indifference and at worst hatefulness and racism. Think back to the resignation one feels when a program crashes and the only option is to click OK. Or the helplessness one encounters when a much needed file is incompatible

1 Ruha Benjamin, Race After Technology: Abolitionist Tools for the New Jim Code (Medford, MA: Polity Press, 2019).

with their current application version. Such episodes not only make for sympathetic anecdotes (including cathartic internet memes), but also illustrate how user-friendliness does not necessarily mean compassion.

A big problem with user-friendliness is its reliance on extracting value. Contrary to popular belief, software is not made for users, it is made for the value users can generate. Whatever kindness we might perceive in these tools is purely transactional. A computer application need only be friendly if it helps users achieve an external monetizable goal, not necessarily an internal one. As a result, software's imagined users must resort to customization in order to cope with the endless process of being monetized either through time or labor. Workers are allowed colorful backgrounds if it means they can stay attentive during Zoom meetings; individuals may turn on dark mode to withstand staring at their screens for longer periods of time. These ostensibly user-friendly features feed directly into Big Tech's play-as-you-work ideology, which, under the guise of compassion, often acts as a trojan horse to extract as much value from individuals as feasibly possible while they play.

More coping mechanism than invited attitude, play in software is a means of dealing with the fact that software is not made for us. Or so the writer and video game critic Ian Bogost claims in his book, *Play Anything*. After describing it as "a way of operating a constrained system in a gratifying way," Bogost goes on to note that play is essentially turning "misery into fun," a way to cope with the banality of the world.[2] In design, for instance, users typically play freely or use software tools incorrectly to achieve novelty. Play, particularly in form-making and representation, at times leads to innovation. But Bogost's kind of play relies entirely on individual resistance or creative freedom that is not always afforded to all. Moreover, to find joy and amusement in the mundane world largely not built for all of us is more of an anesthetic than a radical way of being. Instead of relying on individuals to train themselves into seeing the world as a playground, a more profound position would be to make the world more intentionally playful. To make it a place where we don't have to gamify mundane situations but are instead invited to play by its systems inverts the logic of late capitalism. Is there room beyond the live-work and play-as-you-work models of economic extraction that pervade contemporary life for more purposely playful exchanges? Without compassion and care from a system's designers, play is simply a tactic for enduring that system's indifference to us.

After reading Curtis Roth's call for new modes of self-care, I thought about the potential for emotionally comforting and compassionate software. Although current business models for software companies do not offer room for tools that can heal without turning a profit, current calls for caring infrastructures have gained momentum as a result of the global pandemic. In the summer of 2020, for example, the School for Poetic Computation in New York asked students in their course, Digital Love Languages, "What if all the software we used was made by people who love us?"[3] This provocation suggested a potential fusion between programming and compassion beyond "user-friendliness" that may engender new models for software design. Around the same time, architects in the U.K. expressed their dissatisfaction with the prominent industry-standard software, Autodesk Revit.[4] In a letter that gained notoriety in architectural press outlets, a number of leading international firms painted Revit as highly constraining and a tool with which users must constantly wrestle. If professional software is subject to the whims of a profession, then architecture's current frustration with its relationship to software also hints at a collective desire for alternative approaches.

To shift the perception of professional and productivity software, however, requires a radical re-thinking of

2 Ian Bogost, Play Anything: The Pleasure of Limits, The Uses of Boredom, & The Secret of Games (New York: Basic Books, 2016).

3 Melanie Hoff, Max Fowler, Adina Glickstein, and Amber Officer-Narvasa, "Digital Love Languages," accessed December 19, 2020, https://lovelanguages.melaniehoff.com/

4 Matt Hickman, "Leading architecture firms pen open letter to Autodesk over rising costs, sluggish development," The Architect's Newspaper, July 27, 2020, accessed December 19, 2020, https://www.archpaper.com/2020/07/leading-architecture-firms-pen-open-letter-to-autodesk/

Animal Crossing

ART SQOOL

A Short Hike

professionalization and industry standardization altogether. It may also require developers to imagine tools for small groups rather than universal audiences. Often it is this general applicability and desire for a wide consumer base that pressures software developers to keep adding features with every update, to buy out other start-up companies, and, quite frankly, to make their tools anything but fun.

While it may appear daunting to reverse this mode of thinking, video games offer some insights. In contrast to productivity software, games are built for satisfaction. They shift the location of value from product to experience: a game's value is in its playability. Above all, games provide alternative structures for communication, labor, and creative thinking centered around amusement. Take Nintendo's *Animal Crossing: New Horizons*, for example. Released just as many of us were getting restless at home during quarantine, *Animal Crossing* offered comfort, escapism, and delight in what Ian Bogost called the game's "cute pastoralism."[5] Combining quotidian goals with visual delight, *Animal Crossing* became the ideal quarantine pastime. Sourdough bakers and Peloton riders found in it a means to stay productive. Those who needed an escape saw an alternate reality. And for those in need of routine, it provided delightful rituals such as harvesting fruit or planting flowers. Bogost even went as far as to label the game "a political hypothesis about how a different kind of world might work."[6] More significantly, *Animal Crossing* offered indefinite playability. Much like Electronic Arts's *The Sims*, the game does not end. It also involves intricate labor and goals, so in a way, it is a productivity-oriented virtual platform. Why then does the labor enacted within this virtual world appear far removed from that which we encounter in Microsoft Excel or Autodesk Revit? If building a custom home in *The Sims* requires a similar information dataset as a BIM file and methodically harvesting your fruit in *Animal Crossing* is conceptually similar to plugging data values into spreadsheets, why must one be so much less satisfying than the other?

Video games also provide an alternative economic model for software production. Like music and film, video game production is now split into major and independent developers. The former, referred to as AAA game studios includes figures like Nintendo and Electronic Arts; the latter encompasses small studios and self-publishers. Unburdened by the market obligations of the bigger publishers, indie game developers have produced some of the most experimental and delightful virtual experiences out today, from the visually striking aimless wandering of ThatGameCompany's *Flower* and Adam Robinson-yu's *A Short Hike* to the heartwarming empathy of Popcannibal's letter-writing game *Kind Words* (lo fi chill beats to write to) to Julian Glander's quirky simulation of art school life, *ART SQOOL*. Unlike Microsoft, Autodesk, or Adobe, which seek to monopolize their respective fields, these games stand independently, cater to smaller audiences, and address various perceptions of delight. Removing the economic burden of a vast user base and all-in-one solutions allows software developers to craft bespoke, intimate, and diverse experiences. Something like an indie subset of design software focused on delight could be a radically liberating and compassionate form of care.

Although software that makes us feel good might not sound like a radical proposition, it might be exactly what we need right now. For a start, let's imagine software not simply as a tool, but as an instrument, something that can be tuned and played and enjoyed. Let's also collectively seek out alternative mediums for work where we may share experiences and communicate meaningfully with each other beyond the supposedly neutral sterility of skeuomorphic pages, spreadsheets, CAD layouts, talking head grids, and chat dialogues. There must be

5 Ian Bogost, "The Quiet Revolution of Animal Crossing," The Atlantic, April 15, 2020, accessed December 19, 2020, https://www.theatlantic.com/family/archive/2020/04/animal-crossing-isnt-escapist-its-political/610012/

6 Bogost, "Animal Crossing".

something between productivity software's utility and video game frivolity. As of now these sites are hazy and difficult to find; they might just need to be designed.

Galo Canizares is a designer, writer, and educator. His work blends absurdity, genre fiction, world-making, simulation, and parafiction to address issues in technology and the built environment. He is the recipient of the 2016-17 Howard E. LeFevre '29 Emerging Practitioner Fellowship, and in 2018 was awarded the Christos Yessios Visiting Professorship at The Ohio State University. His writings have been published in various journals and he is the author of Digital Fabrications: Designer Stories for a Software-Based Planet, a collection of essays on software and design published by Applied Research & Design. He co-directs the architectural practice office ca.