Is Software Property?
The Missing Half of the Property Metaphor and Other Better Alternatives

Chao-Kuei Hung
Information Management Department, Chaoyang University of Technology
Taichung, Taiwan

Hilaire Fernandes
Département de l'Instruction Publique,
Geneva, Switzerland

ABSTRACT

The "property" metaphor of software creations demands Internet censorship laws and other laws harming physical property rights, restricts our understanding, and works against attention economy. Other metaphors of software such as fire, public transportation, mathematics, artistic creation, law, wiretapping devices, speech/propaganda, and extension of human nervous system recognize the fact that the duplication or repeated uses of a piece of software incurs practically zero extra cost once it is created. They also provide insights into more efficient possibilities of creating and making use of software for the society as a whole and the dangers of possible abuses of software whose creators are not motivated by money but by some other malicious intentions.

Keywords: Intellectual Property, Attention Economy, Metaphor.

1. INTRODUCTION

In mainstream thinking, Intellectual creations in digital formats such as articles, software, music, and videos are portrayed as properties of the creator to be protected and sold. Not only does this viewpoint conveniently ignore the other half of the property metaphor -- namely its property right when in the consumers' hands -- but also it misses a critical feature of digital contents -- namely the nearly-zero cost of copying such contents. The property metaphor seriously limits our understanding of various uses of digital contents and is especially inhibitive and mind-narrowing when the digital creations involved are software. In this paper we propose several alternative viewpoints/metaphors that recognize the zero cost of copying and emphasize more powerful potentials of software, thereby providing insights to seemingly unrelated phenomena and problems suddenly arising in recent years as the Internet grows to reach more people and the software gradually eats everything.

2. THE OTHER HALF OF THE PROPERTY METAPHOR

The property metaphor only works when it is in the creators' hands. It makes the cloning capability of the Internet and of the computers analogous to the money copying machine for the creators when the software is mainly considered as a product to be sold and profited from as a reward for the creators hard work, even though this fact is seldom mentioned in the "intellectual property" propaganda.

The consumers who have bought the software also have the same money copying capabilities at their disposal and therefore must be prohibited from exercising these capabilities on their devices -- mobile phones, tablets, and computers -- to prevent the inflation of these digital commodities, which would hurt the profits of the creators. Paradoxically, this requires the society to establish a legal system such as Digital Rights Management and the Anti-Circumvention Provision of the Digital Millennium Copyright Act to strip away the physical device property rights of the consumers, as was the wish of RIAA in the recent youtube-dl take down event. [1] At risk as well is the first-sale doctrine, which would otherwise make the property metaphor somewhat believable.

3. THE FIRE METAPHOR

Software is like fire in that (1) it is useful in innumerable different circumstances, (2) it can be easily copied, and (3) it advances human civilization into a whole different level. [2] Taking this viewpoint, the human kind will probably be better off just letting all software be copied freely regardless of the desires and wishes of its creators. Surely this will result in certain software creators no longer incentivised to create more software. Yet it is absurd to assume that the world -- which is gradually being eaten by software now -- will stop producing new software simply because there no longer exists software sales. Certainly there will be paid-jobs for writing software since (non-software) companies and governments will still need software in their operation. Some may decide to share its creation with the world and some may decide to protect it as a trade secret. Still some others may pool together their resources across the same industry to share the cost of producing a piece of standardized software that will benefit the entire industry as a whole. The fire counterpart of all these scenarios most likely happened with our primitive ancestors, who didn't have copyright protection for the fire they laboriously created. There will no longer be any need to censor the Internet or to instill mechanisms such as DRM and DMCA which threaten the physical device property rights of the software users.

("Consumer" is no longer an illuminating term in this metaphor.)

4. THE PUBLIC TRANSPORTATION METAPHOR

Software is like public transportation. You create it once, and with proper maintenance, it can be repeatedly used for a long time by an indefinite amount of people with little extra cost. In fact, the more people use it regularly and report problems whenever they see any, the easier it is for the maintainers to keep it in good shape. Office software and drawing tools are used across all industries. At the same time, they may not be the first priority for the governments engaging in cybersecurity defenses. Therefore there is less incentive for the governments or any single industry to invest money and hire programmers to create them and share them for the benefit of the rest of the world as suggested in the fire metaphor.

Viewed as the public transportation infrastructure such as highways, railways, and airports, however, such software may be best maintained by the government using tax-payer money. Not everyone paying taxes uses these infrastructures equally frequently but it is relatively acceptable considering the prohibitive cost that would otherwise result if these infrastructures were the exclusive properties of profit-making corporations. Not only would such an infrastructure-owning corporation raise the tariffs as high as the society allows it, but also it would create artificial incompatibility with the infrastructure built by other competing corporations to increase its monopoly power. For example, rail widths could be made different. Traffic signs and signals, and thereby drivers licenses could be made incompatible with competing infrastructure builders -- and they could be declared as "intellectual properties" to be protected just so that no other competitors could use similar signs and signals and thereby raising the "barrier of exit" for road users. [3] The builders and owners of these infrastructures might even stipulate that only cars of their own make are allowed to run on their roads. There would be nearly parallel but non-interchangeable highways running alongside each other. Citizens would be forced to choose one system and stick to it for the rest of their lives since the barrier of exit would be prohibitively high once a decision is made. Monopoly would be an inevitable and inescapable result. Thankfully none of these is true with the public transportation systems in our society. The reader can easily translate these ridiculous analogies to their software counterparts, and yet most software users would find them quite natural and acceptable because they are used to the one-half property metaphor.

5. INTERLUDE: FAILURE OF THE MONEY MOTIVATION

The mainstream one-half property metaphor emphasizes money as the sole motivation for software developers to the exclusion of all other possibilities. However, extrinsic motivation may not always be the best driving factor in human behaviors. Dan Pink pointed out in his TED talk that psychologists repeatedly showed the effectiveness of intrinsic motivations over extrinsic motivations in creative tasks while extrinsic motivations work best for tasks with a narrow focus. [4] Writing software is very often more a creative task than a task with narrow focus. It is most obvious to see intrinsic motivations -- such as "developer's personal itch" as Eric Raymond coined it -- at work In many Free/Libre/Open Source Software projects. [5]

The money motivation quite often even works against the original goal of "promoting the progress of science and useful arts" of the United States constitution, which provides legitimacy to present day world-wide copyright laws. There are countless examples of software designs that work against good engineering principles such as interoperability, forward and backward compatibility, and modularity, that are designed to raise the barrier of exit, and thereby create lock-ins from which users cannot easily break free. Truly open file formats such as png and html have their specifications openly available and thus implementable by any competitive entities, commercial or not. Microsoft secured its desktop monopoly via its then-proprietary doc/xls/ppt formats, which were often incompatible even with its own older versions, thereby forcing recipients of the new format to buy the newest version of the software. Similarly, proprietary communication protocols also play a key role in advancing social networking giants' monopoly power today, as opposed to the open protocols such as smtp, pop3, xmpp, and sip of the old days which created a federated ecosystem and a level playing field for a more diverse and interoperable environment with many competing companies and FLOSS projects.

Each of the following metaphors, in addition to providing fresh insights regarding software, also challenges the money motivation.

6. THE MATHEMATICS METAPHOR

Software is like mathematics. The abundance of functional programming languages such as ML and haskell (and lisp to some extent) and declarative programming languages such as prolog drives the point home. In many cases there are only a small number of the most natural and tidy ways of writing a piece of code given a clear objective, and they are typically repeatedly "invented" by untrained young programmers, just like certain mathematical theorems are repeatedly re-discovered by talented youths. That is why Oracle's attempt to hijack the java interface (and thereby the language) using only 9 lines of codes as its copyright weapon looks like an unbelievably bad joke to knowledgeable programmers. [6]

Mathematicians play with mathematics and share their findings with the world not for profit but for the fun and joy of it. Would the world be better off if everyone making use of the Gaussian elimination method is required to pay royalty to Gauss's descendants, or more likely, to whichever international corporation that obtained its copyright? Would there be more or less applications of the Newton-Raphson method towards building useful civil and electric engineering constructs and consumer products, or would there be more legal disputes that prevent the production of such useful things for the society, if the "copyright" of this root-finding method is strictly protected by laws?

7. THE CLAY AND NOTEBOOK METAPHOR

Software can be seen as a media or tool to express ideas, just like crayons, paper, musical instruments, clay, and natural languages are, but with a supplementary dynamic dimension.

Imagine a classroom of kids collaborating on a large clay art project. What if a kid can manipulate and improve his/her own part, plus someone else's with permission, without destroying others' work, even after the parts have been assembled? What if each kid can further make a duplicate of the entire assembled work and modify it to his/her own liking? GitHub is the adult and software version of this scenario, and dynabook is the as-yet unrealized, education version for the kids.

The computer science researcher Alan Kay [7] coined the term dynamic media to describe such computerized media. Those media are constructed with a graphic user interface and operated on a computer of the size of a paper notebook, called dynabook. [8] Considering the limitation of the hardware then, it was a pure conceptual computer only implemented on larger hardware -- the Alto -- large as a small cabinet. It was affectionately named the interim-dynabook. The target users were primary and high school students as they were the most difficult public to make the computers accessible to. Many modern GUI concepts copied this idea from the interim-dynabook, unfortunately only superficially, and with additional artificial restrictions which defeat its far more visionary goal.

When using these modern versions of the dynabook on fundamental learning activities, the right management of the computerized textbook and software may get in the way. Let's say Larry the teacher wants to reuse some textbook materials, with some modifications to fit the contents suitable for his students. With the pen and paper devices, he would produce digital supplementary teaching content based on these adjustments then distribute it as paper documents. Obviously with a true dynabook Larry should be able to digitally copy and modify some of the textbook contents. However, most likely today's computerized textbooks will be provided in a frame to protect the intellectual property right, and Larry will therefore have to produce the adjusted documents from scratch as he does in a pen-paper environment. The computerized scenario is impractical and has little added value, restricting its usefulness in the host environment. However Larry could decide to use third party textbooks distributed under a free license and produced by volunteers with a strong intrinsic motivation, similar to the free software. [9] It is a game changer on reuse and adaptation of the contents.

Then what about the software itself? Would people use it willingly as they use public transportation Can Larry modify some part of the software he uses and share it with his students? One may say that as a teacher, Larry is not skilled enough to do so. However, computer languages are still very new considering the extent of human knowledge history. As of today it is already considered part of literacy in education, and it will likely be a fundamental skill for teachers in a not too distant future.

Today, Larry may not be able to modify the software not because he's not skilled but mainly because it is impractical to do so. Legal restrictions aside, the monolithic and compiled nature of system software such as GNU/Linux pose a challenge to the modification of its source code. This is where the dynabook concept and its practical implementation in the interim-dynabook is still new and largely hardly understood, 40 years later.

Indeed, the interim-dynabook took the original approach where its operating system was written in itself. This means that dedicated tools are there to assist the users willing to study, to modify and to share the source code of the very system of the dynabook itself.

The software Larry will use on his dynabook would then be written like the dynabook system itself. Its source code will sit along the end user application, easily accessible for modification. When Larry uses the dynabook interactive geometry software, instead of proceeding with its mouse user interface, he can decide to parameterize the application with a small 10 lines source code program of his own, to produce a dedicated dynamic media to share to his students' dynabooks. [10,11]

As a consequence of this parametrization dimension of the dynabook applications, when Larry needs to use several software to produce sound teaching contents, he could write a small program to glue these software together and to produce an unique mixed dynamic media. Students could make their variations and contributions as well. As the dynabook showed us 40 years ago, there is no technical reason software could not be shared, modified and even parameterized as easily as other digital contents. Properly implemented in a classroom, its use can be the supercharged version of the clay metaphor.

8. THE SPEECH AND PROPAGANDA METAPHOR

Software is both a form of and a vehicle for speech and propaganda, and is therefore to be protected as well as to be aware and wary of.

In the Perl poetry contest, the Perl programming language is a vehicle for, and the poems written in it are a form of speech. When DVD CCA tried to censor the DeCSS decoder, Rene Hollan's self-documenting steganography effectively posed the question: dare you censor my article, which encodes a piece of free software? [12] So did Molleindustria the game publisher with their satirical title "phone story", to which Apple replied with censorship. [13] The Electronic Frontier Foundation actually established that code is speech by a legal case in which Berkeley mathematician Daniel Bernstein strived to publish his encryption algorithm against the wishes of the USA DOJ to censor it. [14]

GitHub, the renowned programmers' social networking site, has become an important safe haven for Chinese citizens to fight government censorship partly because of its mechanism for backing up and collaboratively improving software. [15] So are blockchains in cryptocurrencies used for this purpose. Let alone the numerous cases of fight for political freedom of speech in various video games by citizens from many countries.

The very raison d'etre of the Xuexi Qiangguo app is to serve as a vehicle for Chinese Communist Party's propaganda. CCP also employ as propaganda Video games such as "Everyone Hit the Traitors" and online games about fighting Japan. [16,17]

The propaganda-and-speech metaphor clearly provides far more insights than the property metaphor in discussing the controversial expulsion of the parler app from both the play market and the app store. Historically the open source community have more resonance with this metaphor and may therefore be in a better position to provide advice in such

conflicts. One case in point is the federated social network mastodon which had to deal with Gab the far-right group. [18]

Default values of software can and have been heavily used as propaganda. [19] Microsoft's tenacious grip of Internet Explorer shipped as the default browser along with its operating system around 2009 was an example of the propaganda metaphor (and the attention economy viewpoint, see below) completely trashing the property metaphor. [20] Apple's "sent from my iPhone" message is also a propaganda successfully popularized by the default settings.

9. CONCLUSIONS

Nobel laureate Herbert Simon pointed out that the overabundance of information results in the scarcity of attention. To take one step further, the authors argue that attention, not information, should be something to be priced and regarded as property if we honor its natural scarcity in the information age. Advertisements represent the most important part of the revenues for both Google and Facebook, the two most outstanding giants that benefit from the zero-cost copying nature of the Internet. Advertisements are payments to buy viewers' attention. As natural as it would be to view air as property to be protected and traded in space or on the Moon, it simply does not make sense to view it as property on earth. This is so obvious for all to see partly because "distance in time and space lends focus" as Isaac Asimov put it. The change of information from a scarce resource to an overabundant one, however, happened in place on earth within a short period of a few decades, and that's probably why we fail to recognize the inadequacy of the property metaphor of software to the extent that mainstream media don't even question the threat it poses on the very tangible users' physical property rights (phones, tablets, and computers) through the controversial DRM mechanism. A good metaphor should make this viewpoint obvious.

Software is like food recipe to be shared and enjoyed. Software is law governing the virtual world as Lawrence Lessig wrote an entire book to explain it. [21] Software is a zero-cost-cloning -- or even negative cost from the developers' point of view -- wiretap device that victims willingly and eagerly -- sometimes with payment -- install on their mobile phones for the wiretappers such as NSA, GCHQ, CCP, as well as Facebook, Google, Apple, and many low profile analytics companies. With the invention of devices to be embedded inside or interfaced with human body such as the neuralink, software is quickly becoming an extension of our nervous system that senses and controls these devices. Or should we insist on treating it as the property of the software creators of these devices inside our bodies?

Different metaphors of a new concept or technology frames, shapes, and also limits how we think about it. Software as a relatively new, and completely revolutionary, technology to human societies has far more aspects to be explored than the failing half-property metaphor can offer. We simply cannot afford having our thoughts and imaginations limited by it.

10. REFERENCES

[1] D O'Brien, "The Github youtube-dl Takedown Isn't Just a Problem of American Law." Electronic Frontier Foundation 2020. https://www.eff.org/deeplinks/2020/11/github-youtube-dl-takedown-isnt-just-problem-american-law (accessed November 18, 2020).

[2] L Sjöberg, "Fire, Work With Me | The Brunching Shuttlecocks" 2020. http://brunching.com/copyfire.html (accessed November 18, 2020).

[3] S McNealy, "Technologys Barriers to Exit." EWEEK 2020. https://www.eweek.com/it-management/technologys-barriers-to-exit (accessed November 18, 2020).

[4] "The puzzle of motivation." 2020.

[5] E Raymond, "The Mail Must Get Through" 2020. http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s02.html (accessed November 18, 2020).

[6] A Verma, "9 Lines Of Code That Google Allegedly Stole From Oracle's Java." Fossbytes 2016. https://fossbytes.com/9-lines-of-code-that-google-stole-from-oracle-java-android/ (accessed November 18, 2020).

[7] A Kay, A Goldberg, "Personal Dynamic Media." Computer 1977;10:31–41. https://doi.org/10.1109/C-M.1977.217672.

[8] A Kay, "A Personal Computer for Children of All Ages" n.d.

[9] Mathematic Teachers Association, "Textbooks & Exercises books,." Sésamath 2020. https://manuel.sesamath.net/ (accessed November 18, 2020).

[10] H Fernandes, "The Newton-Raphson Method." Be a Geometer 2020. https://blog.drgeo.eu/2019/05/the-newton-raphson-method.html (accessed November 18, 2020).

[11] H Fernandes, "La méthode de Newton-Raphson - Les nouvelles technologies pour l'enseignement des mathématiques." MathemaTice 2019.

[12] D Touretzky, "Steganography Wing of the Gallery of CSS Descramblers." Dave Touretzky's Page 2020. https://www.cs.cmu.edu/~dst/DeCSS/Gallery/Stego/index.html (accessed November 18, 2020).

[13] M Brown, "Apple Bans Phone Story Game That Exposes Seedy Side of Smartphone Creation." Wired 2020.

[14] A Dame-Boyle, "EFF at 25: Remembering the Case that Established Code as Speech." Electronic Frontier Foundation 2015. https://www.eff.org/deeplinks/2015/04/remembering-case-established-code-speech (accessed November 18, 2020).

[15] Y-L Liu, "Chinese Users Turned GitHub into a Land of Free Covid Speech | WIRED." Wired 2020. https://www.wired.com/story/china-github-free-speech-covid-information/ (accessed November 18, 2020).

[16] A Schrader, “Chinese game ‘Everyone Hit the Traitors’ lets players attack Hong Kong protesters.” New York Post 2019. https://nypost.com/2019/12/06/chinese-game-everyone-hit-the-traitors-lets-players-attack-hong-kong-protesters/ (accessed January 16, 2021).

[17] “China’s Online Games Designed with Propaganda in Mind.” 2013.

[18] A Robertson, “How the biggest decentralized social network is dealing with its Nazi problem.” The Verge 2019. https://www.theverge.com/2019/7/12/20691957/mastodon-decentralized-social-network-gab-migration-fediverse-app-blocking (accessed January 16, 2021).

[19] K Kelly, “Triumph of the Default.” The Technium n.d. https://kk.org/thetechnium/triumph-of-the/ (accessed January 16, 2021).

[20] “European Commission - Competition - Delivering for consumers - Choice of web browsers for PC users” n.d. https://ec.europa.eu/competition/consumers/web_browsers_choice_en.html (accessed January 16, 2021).

[21] L Lessig, “Code.” Version 2.0. New York: Basic Books; 2006.