GD399.01 Special Topics in Graphic Design: Open Source ran from January to May 2018 at the Maryland Institute College of Art (MICA). We explored new tools and ideologies for making design using the lens of "Free/Libre/Open Source (F/LOS)." Each student was asked to use F/LOS tools in the making of their design projects all semester, and to experiment with how the ideas and technologies from this realm might serve their needs. While using some of the F/LOS software for final outcomes wasn't always possible (we worked on templates for our school library for example, they still needed things in InDesign), the conversations we had and ideas that surfaced from readings and classroom visitors made their way into student work both inside and outside the classroom.

Open Source Design

Why have developers latched onto the open source movement, but designers have not?
— GARTH BRAITHWAITE

A web search for "Open Source Design" returns several resources to start understanding how F/LOS has a graphic design impact. Near the top of the list is *The Open Source Design Manifesto* by Garth Braithwaite, a designer at Adobe. The manifesto is as follows:

I will:
- find opportunities to design in the open
- share my design experiences; both the good and the bad
- find time for meaningful projects
- openly participate in design discussions
- work with other designers by choice
- improve my toolbox

With *the Open Design Manifesto*, and a 2013 talk called *Designers Can Open Source*, Braithwaite explores simple actions/behavioral changes designers can take to become more “open.” This revolves around an increase in sharing: sharing processes, especially the failures; and post as you are working, show how things evolve. This (hopefully) creates a new kind of ecosystem where designers are friendlier and more collaborative with their neighbors — more un-self-concious.

In response to Braithwaite’s ideas, we decided to post all of our work on github and to record all of our discussions. This felt like a good place to begin the term. We had decided to be open, to share more, to start to utilize at least one tool of open source developers (Github), but where are the roots of this way of working?

Linus’s law

The way we think about and talk about open source today owes a lot to Eric Raymond’s *The Cathedral and the Bazaar*. In the essay, Raymond analyzes how Linus Torvalds (and the distributed hacker crew Torvalds helped lead) developed the Linux Kernel. Raymond

outlines and explains nineteen points key to Linux's and Linus's successes. There are two in particular that merge well with Braithwaite's desires.

As soon as a new version of Linux was working, it would be posted for others. This "release early, release often" maxim flew in the face of existing software development practice: one typically did not want to release buggy code, it was feared that users would abandon your project, so one waited until every error was found before releasing. Raymond found that this was not the case with Linux. `[I]t was clear that something different and much healthier was going on [with Linux].` The success was in how Torvalds treats users — they are treated as colleagues and co-developers. Release early and often is a predicate for Raymond's next takeaway from his analysis: "Linus's Law." Linus's law states that "Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone," or "Given enough eyeballs, all bugs are shallow." Get as many users as you can to tinker with your design, and make sure they tell you about everything right and wrong that they find — especially the wrong.

This is what Braithwaite is trying to get designers to understand that developers seem to have latched onto: many eyes make light work. In graphic design we are trying to find the best visual solutions — but how often do we throw out partially formed ideas and see if anyone can poke holes in our thinking early on in the process? This can be done within a classroom or studio between colleagues — as things come together make sure you're passing things around between desks/desktops; or, it can take place out in the world at large (via something like Dribbble? Github? Behance?).

More Than A Theory

It is absurdly arrogant to begin the design process with an empty piece of paper

— HELLA JONGERIUS & LOUISE SCHOUWENBERG

What Raymond is describing in The Cathedral and the Bazaar is not anything novel. There are other structures

enabling cultural creation via distributed means and including bug fixes from the community: vernacular design. What is described by Raymond is not so different from the way that human creative endeavors have come into existence for most of our collective past. In fact “the ‘open source’ way is closer to how human creativity has always worked” — it’s the proprietary model of production that is new.

In *How Buildings Learn*, Stewart Brand dissects vernacular building and describes a process that predates the Linux community while sounding uncannily similar. In Brand's view, the vernacular process is a systematic framework for evolving concepts. A culture steadily culminates frameworks over time. Vernacular designers reuse invented common forms and methods for common, everyday tasks. Vernacular in this context means common designs by common people. Common features survive the passage of time when they are collectively understood as “good.” Over the years then, a vernacular incorporates more and more “good” features while eradicating “bad” ones. Contemporary design typically looks for new or unique solution to a given problem — this is counter-productive to creating traditions and building tested solutions over time, and it cannot easily tke advantage of this methodology.

To illustrate this in *How Buildings Learn*, Brand references the Cape Cod house. His included diagram is named “The Evolution of the Siasconset Whale House.” This chart visualizes what Raymond is trying to explain to us in *The Cathedral and the Bazaar* (Braithwaite also wants contemporary designers to understand this), that simple things released into the wild of a user community will grow, evolve, and change — and will evolve improvements. The improvements merge back into the computer code or architectural language. Repeat. Torvalds is a genius to Raymond not because he uncovered something new, but in that he managed to take an existing form of cultural production and accelerate and decentralize it so as to maximize merging fixes and improvements and features!

If new software can be designed and constructed this way, and vernacular cultural objects have always existed this

way, what's so hard in contemporary visual design?

Free as in Freedom

> I consider that the Golden Rule requires that if I like a program I must share it with other people who like it.
> **— STALLMAN IN GNU MANIFESTO**

Richard Stallman first published the GNU Manifesto in 1985. In it, he described “Free Software,” which was in opposition to proprietary software. Stallman's motivations for this were both pragmatic and utopian. He hoped to maintain the kind of environment expected from academic computer programmers: a vernacular-like model where everyone builds upon everyone else’s work. Stallman also was hoping to fight the tide of proprietary, softwares and tools he felt rising. Free software is predicated on the following freedoms:

```
A program is free software if the program's users
have the four essential freedoms:
   -    The freedom to run the program as you
wish, for any purpose (freedom 0).
   -    The freedom to study how the program
works, and change it so it does your computing as
you wish (freedom 1). Access to the source code
is a precondition for this.
   -    The freedom to redistribute copies so you
can help others (freedom 2).
   -    The freedom to distribute copies of your
modified versions to others (freedom 3). By doing
this you can give the whole community a chance to
benefit from your changes. Access to the source
code is a precondition for this.
       A program is free software if it gives
users adequately all of these freedoms.
Otherwise, it is nonfree. While we can
distinguish various nonfree distribution schemes
in terms of how far they fall short of being
free, we consider them all equally unethical.
```

Free software, as defined by Stallman, is ethical. It has a point of view. Its clear about where it stands. Braithwaite and Raymond do not deal with this in their discussion or anal

lysis of “Open Design” and “Open Source” respectively. Students that were attempting to get a grip on their position as a designer intentionally using design as a political or critical act found something of use here. The software one chooses to use *can* reflect ethics.

In Opposition

The way Stallman positioned GNU was intentionally oppositional to “main stream” software. As GNU/Linux and other F/LOS tools have grown, they’ve maintained an outside, counter-cultural status. Stallman was never against developers making money — the GNU project and Free Software Foundation have always been clear that Free means Freedom, not free meaning price — but he *is* against developers or companies making money by sacrificing or minimizing the freedoms of their users.

Software/OS as a radical act is a fresh take for design perhaps? Anthony Dunne and Fiona Raby in *Designer as Author* propose that designers `develop a parallel design activity that questions and challenges industrial agendas`. F/LOS does this.

Walking into the average design classroom or studio presents one is greeted by an aluminum wall punctuated by glowing fruit behind which everyone has the same “creative cloud” raining down similar outputs from (mostly) the same sets of aesthetic inputs. By replacing one’s computer and software with F/LOSS alternatives a designer immediately engages in Stallman’s *and* Dunne & Raby's critical activities. Designer's can better access the transparency Braithwaite talks about and better actualize the open source/vernacular un-self-concious iterating, improving, and forking; as well as provide a critique of our current situation by embodying alternative social and technical values.

Free Does Not Mean Easy

We have on the one hand a pragmatic approach based in the history of designing — how we as humans created cultural and societal artifacts; and on the other, a critical design approach that intentionally positions itself as

> We need utopian thinking because without it, we are constrained by the tyranny of the possible.
>
> **— Stephen Duncombe**

Or, Random final thoughts/questions...

Forks in the Road

Incovenience?

Just feels like "bad" software because of the given so much of our time to other things already that it complicated or "worse" through really? or have we just Scribus to work, etc. Is it actually harder or more Inkscape or Blender, or was unable to get a feature of at least once with not being able to do something in cited as a barrier to use. Every student came in frustrated That F/LOS tools are hard to install and/or hard to use is

determined).

materials, so choices like Theinhardt for type were pre-(the Library adheres to MICA's brand guidelines for its the typefaces used in our repositories as they aren't open share all manner of our designs, but we could not include tools were not able to be. We were able to collect and Our content and ideals were inspired by F/LOS — but our create everything for them with Adobe Creative Cloud. there were limitations to what we could do; we still had to editing demo with a library staff member). However, Library's director and staff — we even did a wikipedia ideals of Stallman's and F/LOS at large are mirrored by the thinks of itself as an open entity within the school (the with MICA's library on some new materials. The library not the tools of most peers. In our class we collaborated able to collaborate as the tools have changed — these are themselves and be more open to collaboration, one is less While one might adopt more open practices for

thinking, but its a barrier).
the right approach (Stallman has a rebuttal to this kind of your designs available to everyone might not seem like are working in a way that is at odds w\ what the market wants. If you see design as serving business, then making choosing openness, transparency, and sharing designers oppositional from mainstream modern-capitalism. By

2018-05-04, Baltimore, MD
Kristian Bjørnard

The end?

precluded from accessing the techniques and ideas. forever. This means that we as visual designers are not need as humans have been being made in these ways contemporary startup culture. The objects we actually Open Source software's successes are not unique to Source, what was most useful was being reminded that As a course, Special Topics in Graphic Design: Open application learning.

from the tyranny of tools. One can focus on designing over collaborating, sharing, etc. easy. It also relinquishes one tracked and shared with tools that make a workflow of types are open standards. This means files can easily be Another pedagogical advantage to F/LOS is that the file

as possible or "standard."

propose alternate fictions or expand what is understood the tools themselves can reflect non-traditional values and example, or NodeBox or Processing). It is utopian in that from Adobe or Autodesk (the Spiro tool in InkScape for that do not exist in main stream operating systems nor utopian ways. Its pragmatic in that there are F/LOS tools for my discipline. F/LOS allows for this in pragmatic and

As an educator my role is to help create new knowledge has use as a critique of the status quo.

computer software has different ethics than yours" This using the same computer or software as you." and "my Scribus, etc.) a designer is specifically saying "I am not choosing to use this surreal set of softwares (Linux, model useful for critiquing our mainstream culture. By clients or business. F/LOS ideals create an oppositional that can return to its vernacular roots and not just serve start to see chances for design as a social critique; design F/LOS design means? Is that all that's possible? We should design? Fonts, Icons, Templates, Frameworks. Is this what What do you see out there when you look for open source

Colophon

All type used in this section (well, and the whole book!) is F\LOS. The main text block use Noto Serif. The titles are Savate Serif by John Vetter (forked originally from Velvetyne's Savate). Inline quotations are in Noto Mono. Block quotes following headlines are in Cooper Hewitt. Imagery was created in NodeBox. All the SVGs are in the repo containing this book, feel free to use them — they're all GPLv3 licensed — Do what you want with them, but whatever you use them in you must also let everyone else do what they want with that. Except the image on this page, it is from the Internet Archive's Flickr ... It is part of Flickr Commons, my favorite place to look for photos and illustrations that have no known copyright restrictions, or are actually just flat out in the public domain. For the physical books, we collected paper from our department recycling bins. Everything is french folded. If you're adventurous, cut open a page and see what you find inside.

On The Cover

The cover is a risograph print Mazzy Bell kindly printed for us. The illustrations were drawn in InkScape collaboratively on a Raspberry Pi.