Inventing Copyleft

Christopher Kelty, Rice University

This chapter explores one case—a central one—of invention and controversy in the world of Free Software: that of the EMACS text editor and the "General Public License".[1] The GPL is one of the most widespread and important legal objects in the world of software, and increasingly beyond, in music, film, science and education. The story of the controversy is well known amongst hackers and geeks, but not often told, and not in detail, outside these small circles. Moreover attention is rarely paid to the details of context: the ways in which different actors in the controversy mobilize distinct but overlapping interpretations of both IP law, and the technical functions of software, in order to make *moral* claims. One implication of this "ethnographic" approach is to show how the so-called "hacker ethic" can be seen as the *outcome* of such controversy and negotiation, and not something that precedes or determines the actions of people involved.[2]

The software

EMACS is a text editor; it is also something like a religion. As one of the two most famous text editors it is frequently lauded by its devoted users, and attacked by detractors (those preferring Bill Joy's `vi`, also created in the late 1970s). EMACS is more than just a tool for writing text—for many programmers it was (and still is) the principle interface to the operating system. EMACS allows a programmer to write a program, to debug it, to compile it, to run it, to email it to another user, all from within the same interface. It allows users to quickly and easily write extensions to EMACS itself—

extensions that automate frequent tasks, and in turn become core features of the software. It can do almost anything, but it can also frustrate almost anyone. The name itself refers to its much admired extensibility: EMACS stands for "Editing MACroS." Like all such projects, many people contributed to the creation and maintenance of EMACS (including Guy Steele, Dave Moon, Richard Greenblatt and Charles Frankston), but there is a clear recognition that it had "'RMS' [Richard Stallman's handle] chiseled somewhere" on it.[3]

Around 1978, EMACS began to proliferate to different operating systems and user communities, a fact both pleasing and frustrating to Stallman:

> The proliferation of such superficial facsimiles of EMACS has an unfortunate confusing effect: their users, not knowing that they are using an imitation of EMACS and never having seen EMACS itself, are led to believe they are enjoying all the advantages of EMACS. Since any real-time display editor is a tremendous improvement over what they probably had before, they believe this readily. To prevent such confusion, we urge everyone to refer to a nonextensible imitation of EMACS as an "ersatz EMACS."[4]

Thus, while EMACS in its original form was a creation of Stallman, the *idea* of EMACS or of any "real-time display editor" was proliferating in different technical forms. The phrase "non-extensible imitation" captures the combination of design philosophy and moral philosophy that EMACS represented. Extensibility meant that users could make their improvements easily and equally available to all because EMACS had a clever way for users to both add extensions and to learn how to use new ones (the "self-documenting" feature of the system). Thus, the conceptual integrity of EMACS was compromised when it was copied imperfectly. EMACS has a modular, extensible

design that by its very nature invites users to contribute to it and to extend it and to make it perform all manner of tasks—to literally copy and modify it instead of imitating it. For Stallman, this was not only a clever design, it was an expression of his sense of a moral order he knew from the small-scale setting of the AI Lab at MIT.

Not everyone shared Stallman's sense of communal order, however. So in order to facilitate the extension of EMACS through sharing, Stallman started something he called the "EMACS Commune." In a user's manual for EMACS (AI Lab Memo 554, 22 October 1981) Stallman gave a detailed and colorful explanation:

> EMACS does not cost anything; instead, you are joining the EMACS software-sharing commune. The conditions of membership are that you must send back any improvements you make to EMACS, including any libraries you write, and that you must not redistribute the system except exactly as you got it, complete. (You can also distribute your customizations, *separately*.) Please do not attempt to get a copy of EMACS, for yourself or anyone else, by dumping it off of your local system. It is almost certain to be incomplete or inconsistent. It is pathetic to hear from sites that received incomplete copies lacking the sources [source code], asking me years later whether sources are available…. If you wish to give away a copy of EMACS, copy a distribution tape from MIT, or mail me a tape and get a new one.[5]

Because EMACS was so widely admired and respected, Stallman had a certain amount of power over this commune: he was not the only person who benefited from this communal arrangement. Two disparate sites may well have needed the same macro extension, and so users could see the social benefit in returning extensions for inclusion and becoming a kind of co-developer of EMACS. As a result, the demands of the EMACS commune, while unusual and autocratic, were of obvious value to the flock.

The terms of EMACS distribution agreement were not quite legally binding; nothing compelled participation except Stallman's reputation, his hectoring or a user's desire to reciprocate. On the one hand Stallman had not yet delved deeply into the world of copyright, trademark and trade secret, and so the EMACS commune was the next best thing; on the other hand, the state of intellectual property law was in great flux at the time, and it was not clear to anyone, whether corporate or academic, exactly what kind of legal arrangements would be legitimate. Stallman's "agreement" was a set of informal rules that expressed the general sense of mutual aid that was a feature of both the design of the system, and Stallman's own experience at the AI Lab. It was an expression of the way Stallman expected others to behave, and his attempts to punish or shame people, a kind of informal enforcement of these expectations. In the absence of legal threats over a trademarked term, there was not much to stop people from calling their "ersatz" versions "EMACS"—a problem of success not unlike that of Kleenex or Xerox. As time went on, EMACS was ported, forked, re-written, copied, or imitated on different operating systems and different computer architectures in universities and corporations around the world; within five or six years, a number of different versions of EMACS were in wide use—and it was this situation of successful adoption that would provide the context for the controversy that erupted in 1983-5.

the controversy

In brief the controversy was this: in 1983, James Gosling decided to sell his version of EMACS—a version written in C for UNIX called GOSMACS—to a commercial software vendor called Unipress. GOSMACS was the second most famous implementation of EMACS (after Stallman's itself), written when Gosling was a graduate student at Carnegie Mellon University. For years, Gosling had distributed

GOSMACS by himself, and had run a mailing list on Usenet, on which he answered queries and discussed extensions. Gosling had explicitly asked people *not to re-distribute the program*, but to come back to him (or send interested parties to him directly) for new versions, making GOSMACS more of a benevolent dictatorship than a commune. Gosling maintained his authority, but graciously accepted revisions and bug-fixes and extensions from users, incorporating them into new releases. Stallman's system, by contrast, allowed users to distribute their extensions themselves, as well as have them included in the "official" EMACS. By 1983, Gosling decided he was unable to effectively maintain and support GOSMACS—a task he considered the proper role of a corporation.

Though Gosling had been substantially responsible for writing GOSMACS, Stallman felt some propriety for this "ersatz" version and was irked that no non-commercial UNIX version of EMACS now existed. So Stallman wrote his own UNIX version, called GNU EMACS and released it under the same EMACS commune terms. The crux of the debate hinges on the fact that Stallman used (ostensibly with permission) a small piece of Gosling's code in his new version of EMACS—a fact that led numerous people, including the new commercial suppliers of EMACS to cry foul. Recriminations and legal threats ensued and the controversy was eventually resolved by Stallman rewriting the offending code, thus creating an entirely "Gosling-free" version that went on to become the standard UNIX version of EMACS.

The story raises several questions with respect to the changing legal context: questions about the difference between "law on the books" and "law in action," about the difference between the actions of hackers and commercial entities, advised by lawyers and legally-minded friends, and about the text and interpretation of statutes as

they are written by legislators and interpreted by courts and lawyers. The legal issues span trade secret, patent, and trademark, but it is copyright that is particularly central to this story. Three issues were undecided at the time: the copyrightability of software, the definition of what counts as software and what doesn't, and the meaning of copyright infringement. While the controversy did not resolve any of these issues (the first two would be resolved by Congress and the courts, the third is still somewhat murky), it did clarify the legal issues for Stallman sufficiently that he could proceed from the informal EMACS Commune to create the first version of a free software (copyleft) license, the GNU General Public License, which first started appearing in 1985.

Gosling announced his decision to sell GOSMACS in April of 1983. Prior to Gosling's announcement there had been quite a bit of discussion around different versions of EMACS—including an already "commercial" version called CCA EMACS, written by Steve Zimmerman.[6] Some readers wanted comparisons between CCA EMACS and Gosling EMACS, others objected that it was improper to discuss a "commercial" version on the net.emacs mailing list (maintained by Gosling), an activity that should be carried out as part of the commercial company's support activities. Gosling's announcement (April 9, 1983) was therefore a surprise, since it was already perceived to be the "non-commercial" version:

> The version of EMACS that I wrote is now available commercially through a company called Unipress ... They will be doing development, maintenance and will be producing a real manual… Along with this, I regret to say that I will no longer be distributing it. This is a hard step to take, but I feel that it is necessary. I can no longer look after it properly, there are too many demands on my time. EMACS has grown to be completely unmanageable. Its popularity has made it impossible to distribute free: just the task of writing tapes and stuffing them into envelopes is more than I can handle. The alternative of abandoning it to the public domain is unacceptable. Too many other programs have been destroyed that way. Please support these folks. The effort that they can afford to put into looking after EMACS is directly related to the support they get. Their prices are reasonable.[7]

Gosling's work of distributing the tapes had become "unmanageable"—and the work of maintenance, upkeep, and porting (making it available on multiple architectures) is something he clearly believes should be done by a commercial enterprise. Gosling did not consider GOSMACS to be a communal creation, but he did incorporate the work and suggestions of others, contributions that arrived because of his commitment to keeping it free.

"Free" however, did not mean "public domain"—as is clear from his statement that "abandoning it" to the public domain would destroy it. The distinction is important: "free" means without charge—but Gosling clearly intended to be identified as the author, owner, maintainer, distributor, and sole beneficiary of whatever value Gosling EMACS had. "Public domain" by contrast, implied giving up all these rights.[8] His decision to sell it to Unipress was a decision to transfer these rights to a company who would then charge for the labor he had provided for "free" prior to that point. Such a distinction was not clear to everyone—many people considered the fact that GOSMACS was free to imply that it was in the public domain.[9] Not least of these was Richard Stallman, who referred to Gosling's act as "software sabotage" and urged people to avoid using the "semi-ersatz" Unipress version.[10]

To Stallman "free" meant something more than either "public domain" or "for no cost." The EMACS Commune was designed to keep EMACS alive and growing as well as to provide it for free—it was an image of community stewardship, a community that had included Gosling until April 1983.

The disappearance of a UNIX version of EMACS also fed into Stallman's nascent plan to create a completely new, non-commercial, non-AT&T UNIX operating system called GNU ("Gnu's Not Unix").[11] At this point (1983-4) Stallman likely intended to

require the same "EMACS commune" rules to apply to GNU—rules that he would be able to control by overseeing (in a non-legal sense) who was sent or sold what, and by demanding (in the form of messages attached to the software) that any modifications or improvements come in the form of donations. The GNU Project initially received little attention however, often in the context of discussions of AT&T UNIX licensing practices that were unfolding as AT&T was divested and began to market its own version of UNIX.[12]

Stallman's original plan for GNU was to start with the core operating system, the kernel, but his extensive work on EMACS, and the sudden need for a free version led him to start there. In 1984, and into 1985, he and others began work on a UNIX version of GNU EMACS. The two commercial versions of UNIX EMACS (CCA EMACS and Unipress EMACS) continued to circulate and improve in parallel. By March of 1985, Stallman had a complete version (version 15) of GNU EMACS running on the BSD4.2 Version of UNIX. Stallman announced this software in a characteristically flamboyant manner by publishing an article in the computer programmers' monthly magazine *Dr. Dobbs*, entitled the "GNU Manifesto."[13] The announcement caused some concern amongst the commercial distributors—principally because GNU EMACS 15.34 contained code marked "Copyright (c) James Gosling."[14]

The "discovery" was not so difficult since Stallman always distributed the source code along with the binary, but it led to extensive discussion amongst EMACS users of issues such as the mechanics of copyright, the nature of infringement, the definition of software, the meaning of "public domain," the difference between patent, copyright and trade secret, the mechanics of permission and its granting—in short a discussion that would be repeatedly recapitulated in nearly every software and IP controversy in the

future.

The controversy began in early June, with a posting to net.emacs claiming:

> RMS's work is based on a version of Gosling code that existed before Unipress got it. Gosling had put that code into the public domain. Any work taking off from the early Gosling code is therefore also public domain.[15]

This claim, clearly a false one, brought an extensive reply from Steve Zimmerman, the author of CCA EMACS:

> This is completely contrary to Gosling's public statements. Before he made his arrangements with Unipress, Gosling's policy was that he would send a free copy of his EMACS to anyone who asked, but he did not (publicly, at least) give anyone else permission to make copies. Once Unipress started selling Gosling's EMACS, Gosling stopped distributing free copies and still did not grant anyone else permission to make them; instead, he suggested that people buy EMACS from Unipress. All versions of Gosling's EMACS distributed by him carry his copyright notice, and therefore none of them are in the public domain. Removing copyright notices without the author's permission is, of course, illegal. Now, a quick check of my GNU EMACS sources shows that sure enough, a number of files have Gosling's copyright notice in them. What this all means is that unless RMS got written permission from Gosling to distribute his code, all copies of GNU EMACS constitute violations of the copyright law. All those people making such copies, including those people who allow them to be copied off their machines, could each be liable for large sums of money. I think that RMS had better tell us if he has Gosling's written permission to make these copies. If so, why has he not stated this earlier (preferably in the distribution itself) and thereby cleared up a potentially major point of confusion? If not, why

> has he gone ahead and made many, many people liable for criminal prosecution by recommending that they distribute this code without even warning them of their liability? (People who distribute this code would be liable even if they claim that they didn't see Gosling's notices; the fact that the notices are there is sufficient. "Ignorance of the law is no excuse.")
>
> Now, I have nothing against free software; it's a free country and people can do what they want. It's just that people who do distribute free software had better be sure that they have the legal right to do so, or be prepared to face the consequences. (Jun 9, 1985).[16]

Stallman replied the next day:

> Nobody has any reason to be afraid to use or distribute GNU EMACS. It is well known that I do not believe any software is anyone's property. However, for the GNU project, I decided it was necessary to obey the law. I have refused to look at code I did not have permission to distribute. About 5% of GNU EMACS is close to (though quite a bit changed from) an old version of Gosling EMACS. I am distributing it for Fen Labalme, who received permission from Gosling to distribute it. It is therefore legal for me to do so. To be scrupulously legal, I put statements at the front of the files concerned, describing this situation.
>
> I don't see anything I should warn people about--except that Zimmerman is going to try to browbeat them.[17]

Stallman's original defense for using Gosling's code was that he had permission to do so. According to him, Fen Labalme had received written permission (to make use of, or to redistribute is not clear) the display code that was included in GNU EMACS 15.34. According to Stallman, versions of Labalme's version of Gosling's version of

EMACS were in use in various places (including Labalme's employer, Megatest), and that they considered this a legally defensible position.[18]

Over the next two weeks, a slough of messages attempted to pick apart the issues of copyright, ownership, distribution and authorship. Gosling clarified that GOSMACS was never in the public domain, but that "unfortunately, two moves have left my records in a shambles" and he is therefore silent on the question of whether he granted permission.[19] Gosling's claim could well be strategic: if he had given permission this might anger Unipress, who expected exclusive control over the version he had sold; by the same token, he may have approved of Stallman's re-creation, but not wanted to affirm this in any legally actionable way.

Stallman's biggest concern was not so much the legality of his own actions as the danger that people would choose not to use the software because of legal threats. Stallman wanted users not only to feel safe using his software—but to adopt his view that software exists to be shared and improved, and that anything that hinders this is a loss for everyone, thus the necessity of an EMACS commune.

Stallman's legal grounds for using Gosling's code may or may not have been sound. Zimmerman did his best throughout to explain in detail what kind of permission Stallman and Labalme would have needed, drawing on his own experience with the CCA version of EMACS. Meanwhile Unipress posted an official message that said "UniPress wants to inform the community that portions of the GNU EMACS program are most definitely not public domain, and that use and/or distribution of the GNU EMACS program is not necessarily proper."[20] The admittedly vague tone of the message left most people wondering whether they intended to sue anyone. Strategically speaking, they may have wished to maintain good will amongst hackers and readers of

net.EMACS—an audience likely composed of many potential customers. By contrast, if Gosling had given permission to Stallman, then Unipress would themselves have been on uncertain legal ground—unable to firmly and definitively threaten users of GNU EMACS with legal action. In either case, the question of whether or not permission was needed was not in question—only the question of whether it had been granted

However, a more complicated legal issue arose concerning the status of code previously contributed to Gosling by others. Fen Labalme wrote a message to net.EMACS, which although it did not clarify the legal status of Gosling's code (Labalme was also unable to find his "permission" from Gosling), it did raise a related issue: the fact that he and others had made significant contributions to Gosling EMACS, which Gosling had incorporated into his version, and then sold to Unipress without their permission:

> As one of the "others" who helped to bring EMACS [GOSMACS] up to speed, I was distressed when Jim sold the editor to UniPress. This seemed to be a direct violation of the trust that I and others had placed in Jim as we sent him our improvements, modifications, and bug fixes. I am especially bothered by the general mercenary attitude surrounding EMACS which has taken over from the once proud "hacker" ethic -- EMACS is a tool that can make all of our lives better. Let's help it to grow![21]

Labalme's implication here (though he may not even have realized this himself) is that Gosling may have infringed on the rights of others in selling the code to Unipress. A message from Joaquim Martillo confirmed that "these modules contain code people like Chris Torek and others contributed when Gosling's emacs was in the public domain. I must wonder whether these people would have contributed had they known their

freely-given code was going to become part of someone's product."[22]

The general irony of the complicated situation was certainly not as evident as it should have been given the emotional tone of the debates: Stallman was using code from Gosling, based on permission Gosling had given to Labalme; but Labalme had written code for Gosling which he had commercialized without telling Labalme. In turn, all of them were creating software that had been originally conceived in large part by Stallman (but based on ideas and work on TECO, an editor written twenty years earlier), who was now busy rewriting the very software Gosling had rewritten for UNIX. The "once proud hacker ethic" that Lebalme mentions would thus amount not so much to an explicit belief in sharing so much as a fast and loose practice of making contributions and fixes without documenting them, giving oral permission to use and re-use, and "losing" records that may or may not have existed—hardly a noble enterprise.

By July 4th, 1985 all of the legal discussion was rendered moot when Stallman announced that he would completely rewrite the display code in EMACS, "even though I still believe Fen and I have permission to distribute that code, in order to keep people's confidence in the GNU project. I came to this decision when I found, this night, that I saw how to rewrite the parts that had seemed hard. I expect to have the job done by the weekend."[23] And on July 4th, Stallman was able to send out a message that said: Celebrate our independence from Unipress! EMACS version 16, 100% Gosling-free, is now being tested at several places. It appears to work solidly on Vaxes, but some other machines have not been tested yet."[24]

The speed with which Stallman created this final bit of code was a testament to his widely recognized skills in creating great software—not any urgent (legal) threat.

Indeed, even though Unipress also seems to have been concerned about their own reputation, and the implication made by Stallman that they had forced this issue to happen, it was a month before they even responded, after the fact, that they had no intention of suing anyone—as long as they were using the Gosling-free EMACS, version 16 and higher.[25]

Both Stallman and Unipress received various attacks and defenses from observers of the controversy—many people pointed out that Stallman should get credit for "inventing" EMACS, and that the issue of him infringing on his own invention was therefore ironic—others proclaimed the innocence and moral character of Unipress, who, it was claimed, were providing more of a service (support for EMACS) than the program itself. Some readers interpreted the fact that Stallman had rewritten the display code (whether under pressure from Unipress or not) as confirmation of the ideas expressed in the GNU Manifesto—namely that commercial software stifles innovation.[26] On the other hand, latent within this discussion is a deep sense of propriety about what people had created—many people contributed to making EMACS what it was, not only Stallman and Gosling and Zimmerman—and most people had done so under the assumption (legally correct or not) that it would not be taken away from them or worse, that others might profit by it.

Gosling's sale of EMACS is thus of a different order from his participation in the common stewardship of EMACS. The distinction between creating software and maintaining it is a commercial fiction driven in large part by the structure of intellectual property laws. Maintaining software can mean improving it, and improving it can mean incorporating the original work and ideas of others. To do so by the rules of a changing intellectual property structure forces different choices than to do so according to an

informal "hacker ethic" or an experimental "commune." One programmer's minor improvement is another programmer's original contribution.

the context

The EMACS controversy occurred in a period just after some of the largest changes to US intellectual property law in 70 years. Two aspects of this context are worth emphasizing: 1) practices and knowledge about the law change slowly and do not immediately reflect the change in either the law or the strategies of actors; 2) US law creates a structural form of uncertainty in which the interplay between legislation and case law is never entirely certain. First, programmers who grew up in the 1970s saw a commercial practice entirely dominated by trade secret and patent protection, and almost never by copyright; thus the shift to widespread use of copyright law (endorsed and facilitated by the 1976 and 1980 changes to the law) was a shift in thinking that only slowly dawned on many participants—even the most legally astute, since it was a shift in both strategy and statute. Second, the 1976 and 1980 changes to the copyright law contained a number of uncertainties that would take over a decade to be worked out in case law—issues such as the copyrightability of software, the definition of software, and the meaning of infringement in software copyright, to say nothing of the impact of the codification of fair use and the removal of the requirement to register. Both aspects set the stage for the EMACS controversy and Stallman's creation of the GPL.

Legally speaking, the EMACS controversy was about copyright, permission and the meanings of a public domain and the re-use of software (and though never explicitly mentioned, fair use). Software patenting and trade secret law were not directly concerned, but form a background to the controversy. Many participants expressed a legal and conventional orthodoxy that software was not patentable—i.e. that algorithms,

ideas or fundamental equations fell outside the scope of patent, even though the 1981 case *Diamond v. Diehr* is generally seen as the first strong case for patentability.[27] Software, this orthodoxy went, was thus better protected by trade secret law than patent.

By contrast, copyright law had been rare in software. The first copyright registration of software occurred in 1964, and some corporations, like IBM, routinely marked all source code with a copyright symbol. Others asserted it only on the binaries they distributed, or in the license agreements. The case of software on the UNIX operating system and its derivatives is particularly haphazard, and the existence of copyright notices by the authors varies widely. An informal survey by Barry Gold singled out only James Gosling, Walter Tichy (author of rcs) and the Rand corporation for adequately using copyright notices.[28] Gosling was also the first to register EMACS as copyrighted software in 1983, while Stallman registered GNU EMACS just after v. 15.34 was released in May of 1985. [29]

The uncertainty of the change from trade secret to copyright is clear in some of the statements made by Stallman around his re-use of Gosling's code. Since neither Stallman nor Gosling sought to keep the program secret—either by licensing it or by requiring users to keep it secret—there could be no claims of trade secret status on either program. Nonetheless there is frequent concern about whether one has "seen" any code (especially code from a UNIX operating system, covered by trade secret), and whether code that someone else has seen, re-written or distributed publicly is therefore now "in the public domain."[30]

Stallman's strategy for re-writing software, including his plan for the GNU operating system also involved "not looking at" anyone else's code, so that no trade-secret violations would occur. Although it was clear that Gosling's code was not a trade

secret, it was not therefore clear that it was "in the public domain"—an assumption that might be made about some other kinds of software protected by trade secret. Under trade secret rules, Goslings' public distribution of GOSMACS appears to give the green light to its re-use; but under copyright law (a law of strict liability), any unauthorized use is a violation—regardless of how public it was.

The uncertainty over copyright was in part a reflection of a changing strategy in the computer software industry, an uneven development in which copyright slowly and haphazardly came to replace trade secret as the main way of guarding property claims. This switch had consequences for how non-commercial programmers, researchers and amateurs might interpret their own work, as much as that of the companies whose lawyers were struggling with the same issues. Of course, copyright and trade secret protection are not mutually exclusive—but they structure the need for secrecy in different ways and they make different claims on issues like similarity, re-use and modification.

The 1976 Copyright Act introduced a number of changes that had been some 10 years in the making, related to new technologies like photocopier machines, home audio-taping, and the new video-cassette recorders. It codified fair use rights, it removed the requirement to register, and it expanded the scope of copyrightable materials considerably. It did not, however, explicitly include software—an oversight that frustrated many in the young software industry. Pursuant to this oversight, the presidential Commission on New Technological Uses of Copyright (CONTU) was charged with making suggestions for changes to the law with respect to software. It was therefore only in 1980 that Congress implemented these changes, and explicitly added software to Chapter 17 of the US Copyright Statute as something that could be

considered copyrightable by law.[31]

The 1980 amendment to the copyright law answered one of three lingering questions about the copyrightability of software: is software copyrightable? Congress answered yes. It did not however, designate what "software" meant. During the 1980s, a series of court cases helped specify what counted as "software" including source code, object code (binaries), screen display/output, look and feel, and microcode/firmware. Nor did it specify how much similarity would constitute an infringement and this is something the courts are still adjudicating.

The EMACS controversy confronts all three of these questions. Stallman's initial creation of EMACS was accomplished under conditions where it was unclear whether copyright would apply (i.e. before 1980). Stallman did not attempt to copyright the earliest versions of EMACS—though with the benefit of hindsight, the 1976 amendments removed the requirement to register, thus rendering everything written after 1978 automatically copyrighted—registration representing only an additional effort to assert that ownership in cases of suspected infringement.

Throughout this period, the question of whether software was copyrightable—or copyrighted—was being answered differently in different cases: AT&T was relying on trade secret status, Gosling, Unipress and CCA negotiated over copyrighted material, and Stallman was experimenting with his "commune." Although the uncertainty was answered statutorily by the 1980 Amendment, not everyone instantly understood this new fact, or changed their practices based on it. There is ample evidence throughout the Usenet archive that the 1976/1980 changes were poorly understood—especially by comparison with the legal sophistication of hackers in the 1990s and 2000s. Despite the fact that the law changed in 1980, practices changed more slowly, and justifications

crystallized in the context of experiments like that of GNU EMACS.

Second, a tension emerged between the meaning of "source code" and the meaning of "software"—i.e. the definition of the *boundaries* of software in a context where all software relies on other software in order to run at all. For instance, EMACS was originally built on top of TECO (written in 1962) which is referred to both as an editor and as a programming language; even seemingly obvious distinctions (application vs. programming language) were not necessarily always clear. If TECO is a programming language, and EMACS an application written in TECO, then EMACS should have its own copyright; but if EMACS is an extension or modification of TECO the *editor*, then EMACS is a derivative work, and would require the explicit permission of the copyright holder of TECO.

Finally, the question of what constitutes infringement was at the heart of this controversy, and was not resolved by law or by legal adjudication, but simply by re-writing the code to avoid the question. Stallman's use of Gosling's code, his claim of third-hand permission, the presence or absence of written permission, the sale of GOSMACS to Unipress when it most likely contained code not written by Gosling, but copyrighted in his name—all of these issues complicate the question of infringement to the point where the only feasible option for Stallman was to avoid using anyone else's code at all. Indeed, Stallman could have decided simply to unethically and illegally remove Gosling's copyright notice (perhaps under the theory that it was original to Stallman, or an imitation), but he was most likely concerned to obey the law, and give credit where credit was due, and therefore left the copyright notice attached—a clear case of blurred meanings of authorship and ownership. Barring new precedents in court, GNU EMACS 15.34 was the safest option—to create a completely new version

that performs the same tasks, but in a different manner, using different algorithms and code.

Even as it resolved the controversy, however, it posed new problems for Stallman: how would the EMACS Commune survive if it isn't clear whether one can legally use another person's code, even if freely contributed? Was Gosling's action in selling work by others to Unipress legitimate? Would Stallman be able to enforce its opposite—namely prevent people from commercializing EMACS code they contributed to him? How would Stallman avoid the future possibility of his own volunteers and contributors later asserting that he had infringed on their copyright?

By 1986, Stallman was sending out a letter that recorded the formal transfer of copyright to the Free Software Foundation (which he had founded in late 1985), with equal rights to non-exclusive use of the software.[32] While such a demand for the expropriation of copyright might seem contrary to the aims of the GNU project, in the context of the GOSMACS controversy, it made perfect sense. Having been accused himself of not having proper permission to use someone else's copyrighted material in his free version of GNU EMACS, Stallman took steps to forestall such an event in the future—ultimately resulting in the copyleft license and the Free Software movement. Today, Free Software hackers are nearly as deeply educated about IP law as they are about software.[33] Far from representing the triumph of the "hacker ethic"—the GNU General Public License represents the concrete, tangible outcome of a relatively wide-ranging cultural conversation hemmed in by changing laws, court decisions, practices both commercial and academic, and experiments with the limits and forms of new media and new technology.

The conclusion

The rest of the story is quickly told: Stallman resigned from the AI lab at MIT and started the Free Software Foundation in 1985; he created a raft of new tools, but ultimately no full UNIX operating system, and issued a series of licenses, culminating with version 1.0 of the General Public License in 1989. In 1990 he was awarded a MacArthur "Genius" grant, and over the course of the 1990s was involved in various high-profile battles amongst a new generation of hackers—ranging from Linus Torvald's creation of Linux (which Stallman insists be referred to as GNU/Linux) to the forking of EMACS into Xemacs to his frequent participation in—and exclusion from—conferences and events devoted to Free Software.

The creation of the GPL and the FSF are often understood as expressions of the "hacker ethic," but the story of EMACS, and the complex technical and legal details that structure it, illustrate how the GPL is more than just a hack: it was a novel, privately ordered legal "commune." It was a space thoroughly independent of, but insinuated into the existing bedrock of rules and practices of the world of corporate and university software, and carved out of the slippery, changing substance of intellectual property statutes. At a time when the giants of the industry were fighting to preserve and even strengthen existing relations of intellectual property, this hack was a radical alternative that emphasized the sovereignty not of a national or corporate status quo, but the sovereignty of self-fashioning individuals who sought to opt out of that national/corporate unity. The creation of the GNU General Public License was not a return to a golden age of small-scale communities freed from the dominating structures of bureaucratic modernity, but the creation of something new out of those structures. It relied on, and emphasized not their destruction, but their stability.

EMACS is still widely used (version 21.4 as of 2007), the controversy with Unipress has faded into the distance, and the GPL has become the most widely used and most finely scrutinized of the legal licenses. The controversy over EMACS is by no means the only such controversy to have erupted in the lives of software programmers; indeed, by now it is virtually a rite of passage for young geeks to be involved in such a controversy, because it is the only way in which the technical details and the legal details that confront geeks to be explored in the requisite detail. Not all such controversies end in the complete rewriting of source code—and today many of them concern the attempt to convince, or evangelize for the release of source code under a free software license. The EMACS controversy is in some ways a primal scene—a traumatic one for sure—that has determined the outcome of many subsequent fights by giving form to the Free Software License and its uses.

---

[1] Acknowledgements: A longer version of this chapter appears as chapter six of *Two Bits: The Cultural Significance of Free Software and the Internet*, Durham: Duke University Press, 2008. I would like to thank Mario Biagioli for his generous and careful reading of this piece, and Martha Woodmansee, Peter Jaszi, and Adrian Johns for their organization of the conference and this volume. Gabriella Coleman, Hannah Landecker and Lisa Gitelman also provided help and suggestions for which I am grateful. A longer version of this text with complete notes appears in Kelty (n.d). All of the messages referenced here are cited by their "Message-ID" which should allow anyone interested to access the original messages through Google Groups (groups.google.com).

[2] See Levy, Steven *Hackers: Heroes of the Computer Revolution.* New York: Basic Books, 1984; and Himanen, Pekka, *The Hacker Ethic and the Spirit of the Information Age* New York:

Random House, 2001.

[3] Eugene Ciccarelli "An Introduction to the EMACS Editor," MIT Artificial Intelligence Laboratory AI Memo No. 447 1978, pg 2.

[4] Richard Stallman, "EMACS: The Extensible, Customizable Self-Documenting Display Editor" MIT Artificial Intelligence Laboratory, AI Lab Memo 519a, 26 March 1981, p. 19.

[5] Richard M. Stallman "EMACS Manual for ITS Users" MIT Artificial Intelligence Laboratory, AI Memo 554 22 October 1981, pg 163.

[6] Back in January of 1983, Steve Zimmerman had announced that the company he worked for, Computer Corporation of America, had created a commercial version of EMACS called CCA EMACS (Message-ID: <385@yetti.UUCP>). Zimmerman had not written this version entirely, but he had taken a version written by Warren Montgomery at Bell Labs (written for UNIX on PDP-11s) and created a version for programmers at CCA. Zimmerman had apparently distributed it by FTP at first, but when CCA determined that it might be worth something, they decided to exploit it commercially. By Zimmerman's own account, this whole procedure required ensuring that there was nothing left of the original code by Warren Montgomery that Bell Labs owned (Message-ID: <730@masscomp.UUCP>).

[7] Message-ID: <bnews.sri-arpa.865>

[8] The thread starting at Message-ID: <969@sdcsvax.UUCP> contains one example of a discussion over the difference between public domain and commercial software.

[9] In particular, a thread discussing this in detail starts at Message-ID: <172@encore.UUCP> and includes Message-ID: <137@osu-eddie.UUCP>, Message-ID: <1127@godot.UUCP>, Message-ID: <148@osu-eddie.UUCP>.

[10] Message-ID: <bnews.sri-arpa.988>

[11] Message-ID: <771@mit-eddie.UUCP>

[12] e.g. Message-ID: <6818@brl-tgr.ARPA>

[13] Richard Stallman, "The GNU Manifesto" *Dr. Dobbs Journal,* Voulme 10, issue 3, March 1985 URL: http://www.gnu.org/gnu/manifesto.html

[14] The main file of the controversy was called display.c—a version that was modified by Chris Torek appears in net.sources: Message-ID: <424@umcp-cs.UUCP>. A separate example of something written by Gosling bears a note that claims he had declared it public domain, but did not "include the infamous Stallman anti-copyright clause " Message-ID: <78@tove.UUCP>.

[15] Message-ID: <11400007@inmet.UUCP>

[16] Message-ID: <717@masscomp.UUCP>

[17] Message-ID: <4421@mit-eddie.UUCP>

[18] Message-ID: <4486@mit-eddie.UUCP>; Stallman also recounts this version of events in a 1986 lecture at http://www.gnu.org/philosophy/stallman-kth.html.

[19] Message-ID: <2334@sun.uucp>

[20] Message-ID: <103@unipress.uucp>

[21] Message-ID: <18@megatest>. Note here that the use of "Once proud hacker ethic" which seems to confirm the perpetual feeling that the ethic has been compromised.

[22] Message-ID: <287@mit-athena.UUCP>

[23] Message-ID: <4559@mit-eddie.UUCP>

[24] Message-ID: <4605@mit-eddie.UUCP>

[25] Message-ID: <104@unipress.uucp>

[26] Joaquim Martillo, Message-ID: <287@mit-athena.UUCP>: "Trying to forbid RMS from using discarded code so that he must spend time to reinvent the wheel supports his contention that "software hoarders" are slowing down progress in computer science."

[27] *Diamond V. Diehr*, 450 U.S. 175 (1981), the supreme court decision forcing P.T.O. to grant patents on software.

[28] Message-ID: <933@sdcrdcf.UUCP>

[29] Gosling's EMACS version 264 (not the version Stallman copied, which was #84) is registered with the Library of Congress, as is GNU EMACS 15.34. Gosling's EMACS library of congress registration number is TX-3-407-458. The registration date is listed at 1992; Stallman's is TX-1-575-302, registered May 1985. The listed dates are uncertain, however, since there are periodic re-registrations and updates.

[30] This is particularly confusing in the case of "dbx": Message-ID: <4437@mit-eddie.UUCP>, Message-ID: <6238@Shasta.ARPA>, Message-ID: <6447@Shasta.ARPA>, and Message-ID: <730@masscomp.UUCP>.

[31] CONTU Report http://digital-law-online.info/CONTU/contu1.html (visited Dec 8 2006).

[32] Message-ID: <8605202356.AA12789@ucbvax.Berkeley.EDU> Date: Tue, 20-May-86 12:36:23 EDT

[33] see Gabriella Coleman *Social Construction of Freedom*, Ph.D U. Chicago, 2005.