The "Affero" Compatibility Clause

Benjamin Mako Hill For Committee D

May 15, 2006

1 GPLv3 Clause

> 7 (d) They may require that the work contain functioning facilities that allow users to immediately obtain copies of its Complete Corresponding Source Code.

2 Issues

This clause provides for compatibility with code under the Affero General Public License (AGPL) and with similar licenses. These licenses aim to address a situation created by application service providers (ASPs), also known as "web services," who currently avoid the GPL's copyleft provisions by never distributing their derivative works of GPL software. The AGPL, and licenses like it, use the copyright holder's exclusive right to modify software and to derivative works to ensure that *users* of software have access to the source code of their programs. In the AGPL and in similar licenses, this is accomplished by requiring that software not be modified in such a way that it cannot distribute its own source.

Commenters in this round have expressed concern with the compatibility clause in the GPLv3 draft for a variety of reasons. Most of these can be divided broken down into the following categories:

1. Concern that any all restrictions that mandate or prohibit changes in the functionality of a program introduce an unacceptable barrier to the essential freedom to modify

software and that compatibility with any license that mandates *any* functionality is unwise or unacceptable.

2. Concern that the current compatibility clause would allow for compatibility with licenses that introduce unacceptably onerous or impractical burdens for the creators and distributors of certain types of derivative works.

3. Concern that the current compatibility clause would allow for compatibility with licenses that mandate *particular* functionality and, thorugh this mechanism, may also prohibit modification of the work to operate in a substantially differnt way or through a different type of interaction if that modification would also render the former source distribution facilities irrelevant or non-functional.

4. Concern that the ASP issue addressed by the AGPL is too important to be left as an optional compatibility clause and should be integrated into the GPLv3. (This concern is completely incompatibility with the comments in division 1).

5. Concern that licenses implementing the clause could be easily worked around or rendered ineffective.

Comments in each of these areas are discussed in depth in the rest of this section. I have not included redundant comments and have edited comments to include only the most relative sections and sentances and, in some cases, split comments into different pieces.

2.1 Conflict With Freedom To Modify

The strongest opposition to this clause came from groups that felt that the clause impinged on users essential freedom to modify software. Arguing this position, comment 204 states, "this is concerning because it permits a sort of 'invariant section' for programs," and continued to criticize the section in more depth. Another comment expressed a similar sentiment in more depth:

> This requirement would introduce another mandated feature that I cannot remove. A mandated feature that is more than an inconvenience and thus is worse than clause 5c. This option, when exercised, would make the work fail to grant full freedom to modify. ... I would like to see clause 7d dropped entirely, as it allows derivative works to be non-free. – Comment by [807]

It is important to note that while some comments seemed opposed to *any* attempt to address the ASP issue, many, like the comment above, were made by those who were sympathetic to the goals of the the AGPL and similar licenses but who were opposed to the tactic of addressing the issue through entirely through limits on functional modification.

2.2 Major Practical Burdens on Producers

The majority of commenters were concerned that the clause, as written, would allow compatibility with licenses that that would introduce onerous practical burdens for those working on particular types of software or in particular fields of endeavor. The primary example cited in the comments were embedded systems, as illustrated by the following comment:

> This restriction can be incredibly impractical and unsuitable for a large class of programs. Embedded programs, for instance. For most packages in most Linux distributions, the only effective way to satisfy this would be to pack a copy of the source package into the binary package, which is crazy. Imagine this requirement on code in the Linux kernel, glibc, bash; prohibitively impractical. Code with this restriction will be unusable in large classes of projects. – Comment by [420]

Similarly, comment 248 states, "if this option is used, the code can't be used in embedded devices or palmpilots (or equivalents)." Concern of this type seemed to focus on the impracticality of including the CCSC alongside object code in certain certain applications. Embedded systems with very strict space constraints were a frequent good example because it is conceivable that the producer of such a device may be happy to provide source through a method described in section 6 of the GPL but may find it impractical to include source

on the device itself. In this sense, the GPL would be allowing compatibility with licenses that render certain activities prohibitively difficult even though the licensee is acting in good faith and fully in the spirit of both the GPL and the compatible license.

Another commenter argues against subsection 7(d) for several reasons. On the one hand, she argues that by increasing the number of situations where source code was required, the compatibility clause might render certain games less fun:

> A serious bug in subsection 7d. MUDs, MMORPGs, and other online games have the property that both the users and the maintainers do not wish the entire source code to be distributed. This is because, as games, they are more fun to play when it is not possible to cheat by reading the source code. – Comment by [337]

The same commenter argued that the AGPL clause may also lead to an increase in security exploits in free software:

> For security reasons, maintainers of many online systems to which private modifications are commonly made may refuse to use any software subject to a provision such as AGPL 2d ... [because] an intruder who could download the source code might discover programming errors which he could exploit. – Comment by [337]

Of course, the criticism made in both of these comments seem perfectly applicably not only to the clause in question but to copyleft more generally and perhaps to free software as a whole. The committee takes each of the comments in this section seriously but reminds commenters that the copyleft, and the GPL is particular, may introduce pratical difficulties and inconeniences if the FSF and the commmunity determine that these inconveniences are ultimately in the best interests of software freedom for users.

2.3 Clause May Mandate Particular Type of Interaction

In reference to the clause in question, commenter 874 asks, "does this mean the same protocol?" According to our reading of the current language, the clause does not require that

compatible license mandate a particular protocol, technology, or *type of* functioning facility but that licenses that do mandate a particular functioning facility in the service of distributing source code might be compatible with the GPLv3 as written. The following comment, like many similar comments which are not quoted here, explores this possibility in more depth:

> On Debian legal, though, several people expressed concern that code borrowed from a networked program with this type of restriction would become practically unusable if used in a non-networked program (because that program would then have to provide networking capability). – Comment by [790]

Commenters believe that the clause as written would create compatibility with licenses that, by mandating a *particular* functioning facility, would be able to mandate a particular type of use. This might allow compatibility with licenses that bar individuals from making networked programs into non-networked programs or that may be used to tie software to a particular (perhaps vendor specific) hardware platform or software environment. T

2.4 Clause Should Be Mandatory

Some comments suggests that the ASP situation which the AGPL and similar licenses attempt to address is so important that this compatibility clause should be removed and that an AGPL-style clause be integrated as a mandatory piece of the GPLv3. The following comment illustrates this position:

> GPLv3 should REQUIRE developers that modify/incorporate GPL licensed code that source code be available as soon as their code is USED (not: PROPAGATED) outside their organization. Otherwise nearly all GPLed code will be available to proprietary server-side applications as if it were BSD licensed. – Comment by [626]

This class of concerns is incompatible with the first class of concerns.

2.5 Fear of Work-Arounds

There was some concern expressed that compatible clauses could be side-stepped by rendering functioning facilities inoperable. Comment 1098 makes this argument saying, "how is this section affected when the work contains the functionality, but is deployed in such a way that they are inoperative?" A similar, but related, concern was voiced in the following comment:

> This qualifier ("for a work in object code") becomes problematic when this definition is used in section 7d. Imagine a program written in an interpreted language with no object code. If this program's license requires the program include a feature for immediate retrieval of its Complete Corresponding Source Code, the term Complete Corresponding Source Code isn't defined since the program has no object code for the Complete Corresponding Source Code to correspond to. – Comment by [904]

The commmitee believes that both comments seem to hinge on a misunderstanding of the term "functioning facilities." The term "functioning," in the eyes of this committee, implies that the facilities are are, in fact, operational and usable. More importantly, the enforceability of these clauses is not an issue that needs to be considered in the context of this compatibility clause; it is a task for those writing the compatible licenses.

3 Suggestions from Comments

In addition to critiques, many leaving comments made suggestions. Addressing several of the concerns above, the most common suggestion was along the lines of the following comment:

> How about phrasing it as an extension of your distribution obligation? e.g. "They may require that you make source code available to people remotely interacting with the program, just as if you had distributed the object code to them." This way, it's an extension of your distribution obligations and not a restriction on the functioning of the program. – Comment by [204]

This commenter, and others offering similar prescriptions, suggested that the exception be changed so that it could be satisfied through any of the methods in section 6 describing means of providing access to source code along side object code.

Another useful suggestion implying a less significant change is in this comment:

> One way out of this would be to permit this requirement, but only if it delivers the source through whatever medium the user interacts with the program through. – Comment by [790]

The comment argues that the current text could be fixed it if required that compatible licenses be more flexible about the ways in which the source code distribution functionality is created by by mandating any technology appropriate to the software normal use.

4 Recommendations

The GPLv3 clause in question is only a compatibility clause. As such, the goal of this committee is to both to ensure compatibility with certain types of licenses and to block compatibility with others. Although we must imagine what these compatible licenses might say, we cannot suggest new text for these licenses.

This committee does not aim to create recommendations that will ensure compatibility with the current Affero General Public License or any other existing license. In the event of a technical incompatibilities introduced between licenses compatible with this draft but incompatible with future GPLv3 drafts, we hope that the licenses in question can be revised to ensure compatibility with the ultimate released version of the GPLv3.

Mirrored by the suggestions from comments above, this committee recommends one change to the current clause and suggests consideration of a second change or addition to the current list of compatibility clauses.

4.1 Explicitly Prevent Mandating Types of Interaction

The clause should address the concern raised in many comments that the current text would allow compatibility with licenses that mandate a particular type of operation (e.g., operation

over a network) by requiring that the same functionality be retained. Instead, derivers should be permitted to remove that functionality as long as it is replaced with other functionality that is functionally equivalent. Similarly, if a program is changed to operate in a different way or on a different hardware or software platform, licensees should be able to implement, and licensors should be able to require, equivalent functionality appropriate to the derivative works normal interaction with users.

New license text might read:

> 7 (d) They may require that the work contain functioning facilities that allows users to immediately obtain copies of its Complete Corresponding Source Code in a manner appropriate to the derivative works normal interaction with users.

This addresses all comments in the issues subsection on the draft text's compatibility with licenses that mandate a particular type of interaction and, additionally, helps address or mitigate some of the the practical difficulties that conforming to compatible license texts might require.

4.2 Consider Extending Compatibility To Similar Licenses

Other committee members have argued strongly in favor of the suggestion in comment 204, quoted above, that calls for compatibility with licenses that require distribution of source to users through any of the methods described in section 6 in addition to the restrictions on functional modification. Some members of the committee have argued in favor of such a compatibility clause in place of the current language.

The rationale document does not describe why the FSF chose not allow for compatibility with licenses that extend the section six requirements to users of software but the committee has discussed two possible rationales:

1. That limits on functional modification, unlike grants of section 6 responsibilities to users, lie fully within the domain of the exclusive rights of the copyright holder and would ensure compatibility with licenses that, like the GPL, are not contracts.

2. That lacking a widely held legal definition of *user*, limits to licensees functional modification avoid the need to define users in the text the exception.

While the committee remains divided, it suggests that the FSF consider adopting the language suggested above while adding a new compatibility clause that extends compatibility to software under licenses that may require distribution of CCSC through either functioning facilities or any of the methods in section 6. Example text for such a clause might be:

> 7 (x) They may require that when the work interacts with users, but those users can not receive a copy of the complete corresponding source code through normal interaction with the software, you must satisfy the requirements of section 6 ("Non-Source Distribution") as though you had distributed the work to those users in the form of Object Code.

The committee realizes that this suggestion remains impractical without an adequate definition of user. The committee is not, at the time of drafting this report, confident in any suggestion for such a definition. The committee encourages the FSF to FSF should consider possible definitions of the term and to look to other licenses, both free and proprietary, that have attempted to define users for similar purposes. The committee suggest that users should be defined so as not to include those interacting with software solely for the purpose of authentication.