Intentional Free Software

Prolog

There is a truism about travel: when one is traveling for enjoyment, it is the journey getting there which often matters most - it is the journey which makes the trip a full and fulfilling experience. Early master practitioners of software development also understood this: “the journey is the destination”. When Donald Knuth wrote his masterwork, he titled it: *“The Art of Computer Programming”*, not: “How To Get It Done Using A Computer.”

Argument

There have been any number of attempts to categorize the activity of computer programming. Early-on “Art” was a strong contender. Later, this perception was challenged, or confused (ironically just as Free Software was becoming popular.) Much of the misdirection can be attributed to the underlying premise of “Open Source”, which is this: Software, instead of being recognized as a medium of human expression, is classified as a type of “property”. When software code or executables are “property”, then the writing of software is simply “production”. However, if software is seen as Art – then it becomes a very different thing.

Software, once it is viewed in all seriousness as an Art, like Speech, then the expression and use of software becomes understood as a full-fledged cultural entity, where participation with software becomes part of our proper rights as human beings.

Now, if computer programs are a “Medium of Art”, much like literature or song, a cultural entity, we all have a Right to express ourselves in this way, and to share these expressions with others. Sadly, the Free Software movement has neglected to take this notion where it should lead. Even the notion of “sharing” is somewhat misleading as it can take us back to the idea of software as “property”.

OK, Now What?

OK, so how do we advance any human right where that perspective appears to be faltering? We know that Free Software has a specific Free Software Foundation definition, and we know that we are after something closely related to that definition. I think *the way to direct our path is to realize that the very means of creating software are the ends and the embodiment of our intent, in the fullest sense possible.* In this way, Free Software becomes “Intentional Free Software”. Once we internalize this understanding, the way forward becomes clear.

It is now better understood that in order to free the software user from unwanted restriction and conscription, Free Software must be intentionally designed and constructed from the ground-up to be Free Software. Freedom starts with a design for Freedom, carries through with Freedom in the practice of development, and then enlists a license requiring Freedom going forward in sharing development. The essential characteristic is transparency throughout.

One may argue that getting more “free” programs done in some period of time would deliver more freedom, or that the cost of spending more time on each program would work to prevented achieving some degree of freedom. These are reasonable points, but in this essay we present how “Intentionally Free Software” is the standard of freedom we must attain in order to sustain our freedom over the long term in an inclusive cultural context.

Fundamentals

This all sounds very serious… but that does not mean our activities here cannot also be fun. In fact, when software is seen as an art form rather than a commodity, enjoyment and fulfillment are enhanced considerably. If software is constructed from the ground-up to be Free, this means a number of things:

#1. Intentionally Free applications (and systems) will be accessible to the user! The design will be as modular, and as robustly simple as possible. Transparency to any interested party must be paramount. Knuth went as far as to actively advocate for “Literate Programming”, where the main intention is to treat a program as literature understandable to any interested person who should pick it up for study.

#2. Most software today has many dependencies, and to include them all in a package such as a flatpac “app” is not only wasteful, it becomes a net of entrapment where the user has little-to-no control over what he is installing. The solution is to begin with an intent: The first intent is to recognize Stallman’s Four Freedoms for Free Software. And then there is also one more Freedom we must observe: The user will be neither restricted nor conscripted - we will not be forced to use features we do not want!

This strongly implies a modular design, and a design with serious intent to minimize dependencies. If a user/developer does not want a feature, they should be able to safely delete that component, and the program will function as designed – minus that component. Furthermore, it will be relatively easy to fit *new* features into modular, accessible software.

Applications should be more like “programs”. A number of related smaller applications rather than a monolithic behemoth will, in and of itself, make for a simpler, more successful intentional Free design. Static compilation should be seriously considered because in this way each user will be enabled to run applications even when they are not necessarily supported by the system libraries included with the OS.

#3. Unnecessary and opaque dependencies are to be considered a bug and the replacement of modifiable code with binary Blobs a severe bug. Blobs do not represent “modularity” - instead they implement developer takeover and can often hide security flaws. This practice is simply unacceptable in Free software. Blobs directly work against Free Software as Free Speech, taking the censors pen to code and actively preventing people from interacting confidently with associated code.

#4. Intentionally Free Software will be developed with “ease of modification” as a primary design goal. Freedom means you are enabled to understand, modify, and share code. We should forget about personally or organizationally “owning” our software (the “product” notion), because the “Free” in Free Software becomes a reality when the software code itself as well as the executing program is a fully functioning medium in the community. Please note that this is not really a “moral” standpoint: You and I, we WANT software to be Free in practice - thus we make it so. Legally we make our

software Free by enlisting a Free Software license. The license in a strict sense is external to the Freedom inherent in Intentionally Free Software, but given today's world a necessary part of the process.

#5. And sadly, Intentionally Free Software will have to actively distance itself from corporate funding. Corporations are not going to stop trying to control software - they make money from software by making it less free. We will always need to keep our endeavor at some far distance from them. People will argue that it becomes almost impossible to develop significant software without corporate support - but we see instead it is more true that we will be able to develop accessible, useful, and artistically beautiful Free software without the dependencies and strings that corporate money brings. *In fact, well designed and well-crafted Intentionally Free Software will make it easier to build and maintain significant software applications.*

In Conclusion

Software designed and created as advocated above will take software freedom to another level entirely-- where software "freedom" is NOT dictated by corporations (or corporate-captured non-profits), and not hampered by incompatible designs. With Intentional Free Software as a flexible grassroots movement, people will more freely join, leave, and fork projects as they please, the software being transparent to all. Intentional Free Software freedoms will engage with more people, enabling them in their work; and Free Software will become the visible, vibrant, significant Art that it was always destined to be.

Thomas Grzybowski