Understanding Commercial Agreements With Open Source Companies

By Amanda Brock

According to ZDNET's Dana Blankenhorn, most open source software no longer comes from open source developers, but from the proprietary companies. Having worked for 5 years as Canonical's General Counsel, heading up the legal team commercialising services associated with the Ubuntu operating system,[102] I was immersed in the hard work that comes with the commercialisation of open source software and been exposed to the models and tribulations of making money from FOSS. In this article I have taken some time to think about commercialisation as an issue and the models for commercialisation.

Collaboration and sharing are at the heart of open source development. The developers I have met - and I have met many - are in the main a beautifully idiosyncratic, highly intelligent and interesting group of free thinkers. Their inability to accept the parameters of current thinking seems to me to be key to the creativity they demonstrate in the technical solutions they develop. Combined with their idealism, it forms the basis of the thinking that allows them to challenge the logic of laws, such as IP and in particular copyright. Copyleft, developed as part of this free thinking, is a play on the legal term copyright and a great example of this.

---

[102] Amanda Brock is no longer part of the Canonical team. She has left her much loved Ubuntu to become a Director of the international technology law firm, Origin, www.origin.co.uk where she is able to support multiple FOSS developers and companies providing legal and strategy advice.

When I sat down to write this article, I looked online, as I always do. I am without doubt a victim of the Wikipedia culture. Having grown up without a computer and started my working life without the Internet, I am to this day, humbled by the vast sources of information sitting on open source software, in a technology based on that thinking. I can access this information from almost anywhere in the world. Information that, without thinkers like Richard Stallman, people whose motivation is not necessarily fame and money, but who are interested in making a better society through collaboration and the tools they can offer society, wouldn't be there for me or my research. Like people, whose strengths, if overplayed, become their weaknesses, the Internet may not always be a great thing, Wikipedia can be dangerous. How often have you looked up a sniff or sneeze to diagnose yourself with a serious illness.

In searching the terms Stallman and Copyleft on Google's search engine (which runs on server farms driven by open source software) I found a page[103] that Stallman had written for the GNU operating system.[104] A few things he says there really strike me as important. His opening lines, "Every decision a person makes stems from the person's values and goals. People can have many different goals and values; fame, profit, love, survival, fun, and freedom, are just some of the goals that a good person might have. When the goal is a matter of principle, we call that idealism. My work on free software is motivated by an idealistic goal: spreading freedom and co-operation. I want to encourage free software to

---

[103] http://www.gnu.org/philosophy/pragmatic.html

[104] http://www.gnu.org/, GNU is a Unix style operating system based on free software.

spread, replacing proprietary software that forbids cooperation, and thus make our society better."

Stallman refers to free software not open source. He is the father of the ideological free software movement. He goes on to say, "All code added to a GPL-covered[105] program must be free software, even if it is put in a separate file. I make my code available for use in free software, and not for use in proprietary software, in order to encourage other people who write software to make it free as well. I figure that since proprietary software developers use copyright to stop us from sharing, we co-operators can use copyright to give other co-operators an advantage of their own: they can use our code."

Not all code in the world of free and open source software is copyleft. The term is not enshrined in law, but a pun from this man, this free thinker. Its symbol is a reverse of the copyright symbol ©. For those not so familiar with it, "Free software" means software that respects users' freedom and community. Roughly, the users have the freedom to run, copy, distribute, study, change and improve the software. With these freedoms, the users (both individually and collectively) control the program and what it does for them." The GNU web site goes on to say ""free software" is a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech," not as in "free beer"." The essence of "free" software is enshrined in the concept of Copyleft.

As a concept, copyleft takes copyright (where the creator of a work is the owner of the copyright in code they develop and third

---

[105] GPL – the GNU General Public License, is an OSI approved standard open source license published by the Free Software Foundation ("FSF") on the terms of which free software may be distributed by anyone who meets the free software definition and complies with the license terms.

parties may use it only with the owner's consent) and turns it around. It does so by requiring that both the work created and licensed for free use and any changes made to it by a third party are subject to the same ongoing freedoms and that this freedom cannot be restricted without the owner of the original copyright works consent.

To make this concept a reality Stallman and his advisers developed a tool in the licence of free software, which is what most lawyers are referring to when they say that a software licence is copyleft. The copyleft tool is simply a licence provision that ensures that any user who takes and modifies the code they received under a free licence will, if they distribute the modified code, do so under the same free licence without any additional restrictions or conditions and will release the source code in what is distributed under those original terms. The GNU General Public licence is probably the best known and most used example of this type of licensing.[106] Each party will own the copyright in what they have created but must license any new or modified version of the code under the same terms as the code they received, i.e. the licence being copyleft, the rights are left in the free code and the quid pro quo for usage is that new developments must be contributed back.

The licence terms allow anyone to modify and extend the code subject to this restriction included by the guardians of free software for the user's own safety. It intends to preserve the

---

[106] GPLv2 section 6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

freedom under which the software was initially distributed through the use of copyleft.

As the creator of the original work owns the copyright in his work, despite the work being licensed under a copyleft licence for use by others, should any user not contribute back under the same licence, then they would have stepped beyond their rights under the licence and potentially be in breach of the original copyright holder's rights.[107]

Copyleft code when combined in certain ways with other code may create a new collective piece of code which is in fact a derivative work. It's like cracking eggs to make an omelette. Once the eggs are combined they become one new thing not individual eggs. That doesn't mean to say that there are not ways of putting together a new meal with the eggs without mixing them and in which the eggs stay as individual eggs, like gammon with a lovely fried egg on top. But, once they are mixed together to make an omelette, they are combined to create a new and distinct thing, like a derivative work.

Where this happens the new derivative work, like the original copyleft licensed component, will also be subject to copyleft terms. This impact of copyleft has been referred to as a "viral effect", where the free copyleft software, if combined with other code to create a derivative work, "infects" the other code with its freedom. It is something which has created considerable fear in the minds of owners of proprietary code and is perhaps the biggest single factor that dissuades some business usage of free

---

[107] GPLv2 section 4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

software. In short the owners of proprietary code are scared to use free software. They are afraid that the copyleft code will virally effect their own code.

Copyleft code is therefore seen to be subject to restrictions around how it may be distributed and how it may be combined with other code.

Clearly this sits well within the ideology Stallman espouses. His copyleft ideology creates a cascade of free software and freedom. As a paternalistic concept, it protects users from themselves in the interest of freedom. The consequences of copyleft are not an accident, but a carefully crafted attempt to preserve this freedom and remove temptation.

This ideology is akin to political thinking as opposed to software coding. The decision to create this was not a technological or development decision, but the expression of a belief system. That belief system is the centre of free software and the free software movement.

Its principles offer little space for commercialisation of free software. Freedom does not however necessarily mean that no price can be charged for the code and states in GPLv3, in section 4 which deals with distribution or conveying software, "You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee."

The term open source was adopted in 1998, by a group on the US's West coast. The adopters formed a movement that is younger than free software and rejected some of the free software philosophy. Eric Raymond and Bruce Perens founded the Open Source Initiative later that year.[108] Its foundation was, to some extent at least, a reaction against the values of free software.

---

[108] http://opensource.org/history

Like free software, open source software involves the distribution of the source code in software. This means that the human readable secret sauce that allows people not machines to understand and read software, is freely available with the binary computer readable code in any software distributed in this way. This allows users to access the functionality of the code as you would with software distributed on a proprietary basis and also gives access to the methodology of the code that is seen in the source component. It allows recipients of the code to modify, extend and maintain the code themselves, if they have the appropriate skill set. This is different from proprietary software distribution where the source is secret and is akin to free software distribution where the source is shared. This ability to allow software developers to work together on code is the collaboration at the core of open source.

In the same way as the FSF have a set of criteria to be adhered to in the definition of free software, the Open Source Initiative ("OSI") created a definition that sets out 10 criteria to be met to be able to call software "open source". Like free software there is no legal constraint on anyone calling their code either free or open. However, if you wish to comply with the true sense of either meaning then a distributor should meet the criteria set out by either the FSF free software definition or the OSI's Open Source Definition ("OSD").

There are differences in the two groups' criteria and definitions. These differences are based on the ideological differences between the two groups. Open source may be considered to be a more pragmatic approach to software freedom, one that is more acceptable to business and which focuses less on ideology and more on pragmatism. By 1999 the OSI had approved its first list of licences as meeting its OSD. Today the OSI has approved a large number of licences. Its licence review

process and the OSD have become an industry standard for open source licences with the OSI taking on the mantle of a standards body. In its work over the years, the OSI has tried to keep this simple and avoid licence proliferation[109] by reducing the number of licences and categorising licences into groups.

Within the OSI's mission statement it says "Open source is a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in". A commercially focused statement, geared to attract business to open source, by setting out a list of some of open source's attributes and the associated business benefits. A far cry from Stallman's free software ideology! This lacks the paternalistic approach of protecting both freedom in code and users from themselves.

Whilst there are clear ideological differences there are not so many practical differences between the two groups. One of the key practical differences is the existence of permissive licensing through the OSD. A permissive licence is one that has few requirements around how the software licensed under it may be distributed and allows wide usage of the code. It allows distribution on an open basis but does not require the copyleft or share-alike contribution back of modifications and which may allow for modified versions of the code to be distributed under a

[109] The name given to the large volume of approved licenses for open source software which had been approved by the OSI, some of which were neither popular nor frequently used, which had terms which might not be compatible with other approved licenses and the volume of which caused confusion.

different or modified licence. Examples of licences that are permissive include the Apache,[110] BSD[111] and MIT licences.[112]

Permissive licences, such as the BSD do not require derivative works to be distributed under the same licence terms as the original code and allow additional or different licence terms to be added to the modified version of the code. Whilst they require an acknowledgement of the author in the code's notices they do not meet the enforced protection on long-term freedom Stallman sought in his copyleft approach. The width of the licence means that the user has choices that may, arguably, give the individual more freedom in how they use and distribute the code, but which remove the collective freedom enshrined in the copyleft principle. The FSF is concerned that permissive licensing does not protect freedom.

In the case of BSD distributed code, the code can be re-distributed under other terms and could even be taken into proprietary code without infringing the original licence of the code. As the code can be incorporated into a copyleft piece of code, it is GPL compatible and so can be combined with software meeting the free software definition. However, code released under the GPL could not be combined into a BSD licence as this would have the effect of changing the licence and be a breach of the GPL licence. Where code is copyleft, a licence cannot be

---

[110] http://www.apache.org/licenses/

[111] http://en.wikipedia.org/wiki/BSD_licenses

[112] http://en.wikipedia.org/wiki/MIT_License

changed without the consent[113] of the creator of that code. A significant problem in the existence of the two groups is that not all permissive licences approved by the OSI may be suitable for combination with copyleft software licences.

Ideologically the free software and open source movements see themselves as being quite far apart. From the outside looking in this may be very confusing and even seem a little nonsensical. Both movements distribute software that has freely available source code. The difference to any user would lie purely within the terms of the licence chosen for the software's distribution. The user may well not care what the licence terms are. They will only really be interested if the terms have any impact on them. Only users who play with the code by modifying it or combining it with other code are likely to care what the licence terms say.

So, at the start of this chapter I said I would write about the commercialisation of FOSS (free and open source software). Why am I hung up in history? As I started to think about the issues and explaining how I see them I realised that the only way to fully explain them is to work through this thought process. So, understanding where the players in FOSS have come from is important.

I suggest that having thought about the legal and ideological context, we look more generally at how organisations make money from FOSS (Free and Open Source Software). In both definitions a fee can be charged, so there is nothing to stop you

---

[113] GPL section 10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

there. Ideologically, commercialising free software, may not sit well with the FSF and Richard Stallman's ideological goals, but it is not prohibited. Why however would anyone pay for software that they can generally receive free as in beer not just in speech? The answer is that they are likely to save their money for that beer and instead download the software for free. So, what is it that they might pay for? A very good question.

The question's ultimate answers may in fact have been delayed by the good intentions and philanthropic actions of a few large companies and wealthy individuals who have recognised the potential for the greater good in open source and so have offered massive resources to the infrastructure both through pure financing and also through their contribution of employee time and business assets such as patents. IBM is a major contributor to open source. It was a founding member of the Open Invention Network ("OIN")[114] patent pool. The patent pool is free to license and offers licensees the protection of not being sued by patent contributors and fellow licensees who offer cross commitments not to sue on their patents where those patents read on software that falls within the OIN Linux definition. Their contribution to open source has been massive and this type of generosity has played a significant part in the sustainability of FOSS over a number of years. As the market matures, however, so must the funding.

The simplest and most recognised answer to that question in a young market is of course the provision of related and specialist services. Just because the software is free doesn't mean that it is guaranteed. Many FOSS licences expressly state that the software comes without warranty. So, if you want to ensure that the software works either generally or in a specific context, you may

---

[114] http://www.openinventionnetwork.com/

wish to engage the services of a software engineer or a software engineering company to work on that code. They may check it, modify it and develop it further for you. If the code is copyleft, you may have to share your changes if you distribute the code. If you keep it for internal usage you may not have to. If the code is on a permissive licence, you may be able to keep that code or even distribute it without resorting to the original licence terms.

Free software purists may not be happy with your actions as they do not sit well with the ideology of free software with changes being contributed back for the greater good, but there is nothing to stop you keeping the code in house and benefiting from the free or open source software code it was based on. There are many examples where companies, even those considered good citizens in the world of open source have done so. They have utilised freely distributed code, modified it for their own business purposes, kept the modified code for their own usage and received huge cost benefits in doing so. As they do not distribute the modified code copyleft requirements do not kick in.

As server based services such as cloud computing or big data may not involve the distribution of code but the utilisation of services from servers running FOSS code, many companies have benefitted from FOSS software to build their server side services and now sell those services to third parties without passing on the code itself. In this way they may sell highly profitable services based on FOSS software without making a contribution back or serving Stallman's greater good.

As well as buying development services from open source software developers the code can be commercialised server side. As individuals and businesses use these services, purchasing them through an on line process in many cases, we have become increasingly used to contracting for these on a click through basis. This is creating a market place where users may not properly

review the terms and conditions that they sign up to. For providers of these services, to ensure that the terms are enforceable its important to ensure that they are fair and fall within the legal parameters of what can be excluded. Where exclusions need to be drawn to the end user's attention this should be done carefully. There is also a risk that the infrastructure utilised is so closed down that even for bigger contracts the opportunity to negotiate is in fact removed. If this is the case the providers may be storing up long term problems for themselves depending on how much they try to exclude, certain exclusions being potentially contrary to legal requirements with respect to fairness if they are not open to negotiation.

Commercial use of the code may be a little more difficult where the mode of commercialisation is device rather than server based. In this instance, the code may be freely downloaded and utilised on a consumer or business device. The code may be pre-loaded onto the device or downloaded across the airways via a repository on boot of the device. Updates are generally provided in this later way via repositories. The code is still warranty free and may however not be suitable for either a particular device or particular usage of that device without some work. That development work may be done in house or provided by a third party and may be subject to a requirement to contribute the development back depending on the licence type, i.e. copyleft.

The consumer focus of certain devices may also create concerns in larger organisations where there are cascades of existing consumer terms and processes that may not be compatible with every FOSS licence. In these environments it may be necessary for the organisations to adopt licence preferences in their Open Source Policies and to check that the licences in the software coming into their organisations for onward distribution meet their compliance and policy needs.

However usage on the client side where the user interface ("UI") of the code is visible, as it is with software loaded on end user devices as opposed to servers, may encounter one of the more current and controversial legal issues in FOSS, trade marks. Where a brand or trade mark is associated with FOSS software, whilst the code may be freely used, whether and how the brand or mark may be used by a third party is dependent on the owner's trade mark policy. Some brands may require payment for a trade mark licence, others may require that the code is used as is or the brand removed. The owners of trade marks are within their rights to do so and commercial usage which will involve brands and marks should be carefully considered as there may be both a brand usage and cost consideration in doing this. It is of course possible to remove the brand or trade mark from the code and to distribute the code. In some instances that could provide a solution, but for other software, part of the reason for using it will be the user confidence created by the brand and it would certainly be detrimental to remove that brand or mark which inspires user confidence.

In some instances the code developer or sponsor may own other intellectual property rights such as design rights or design patents that will be subject to similar provisions. Again, this is relevant only in devices where the branding is seen, i.e. client software and not on the server side.

In addition to development services, software may require ongoing support and open source software companies will frequently offer to sell such support services. These services vary from being a pure service provision where the updates to the software are already provided free of charge, to a subscription model where the updates are provided only where support services have been purchased.

These add-ons to open source software are now almost a traditional route to the commercialisation of open source software. The ability for companies to make money from these depends on many commercial factors such as the popularity of the underlying software. Red Hat has become very well known as one of the most successful open source companies and in 2012 became the first open source company to generate over $1billion in annual revenue. Their model includes providing support, training and integration of their Red Hat Enterprise Linux ("RHEL") software. This is a subscription model where updates to RHEL are available only via this subscription. Their Fedora operating system offers a free community based alternative to this and much of the Fedora development may make its way into the RHEL product.

The trade mark concern has raised its head with RHEL. CentOS provides a free operating system with its upstream distributor, RHEL. CentOS are able to take the freely available RHEL distribution and use the source code to create an operating system which is very similar to RHEL. Notably the RHEL trade mark is not utilised in the CentOS distribution.

1Red Hat is a leader in this pack. It's the first open source company to generate an annual revenue of $1billion and whilst many open source companies would like to snap at its heels, this is a difficult proposition.

1Google and other search engines that generate revenue from advertising also play a part in the commercialisation of open source software. The revenue generated in their advertising deals not only provides these organisations themselves with revenue but this revenue is shared throughout an ecosystem of toolbar distributors and browsers who contribute to the traffic flow. This form of revenue can be very lucrative to open source companies

that distribute code client side and is very dependent on volume of users.

We live in a time and world of disruptive forces in technology. The Android mobile phone operating system is a good example of market disruption being caused by open source software.

The late Apple founder, Steve Jobs' biographer Walter Isaacson has quoted Jobs as saying ""I will spend my last dying breath if I need to, and I will spend every penny of Apple's $40 billion in the bank, to right this wrong," Jobs told his biographer. "I'm going to destroy Android, because it's a stolen product. I'm willing to go thermonuclear war on this." And so began one of the biggest battles of my life time, the Android patent wars. They are easily the subject of a book and I don't want them to dominate this article. So, I will keep it short. Apple and Microsoft have alleged that the Android operating system infringes various patents held by them. They have attacked makers of android devices in multiple patent suits in various jurisdictions. The free nature and popularity of the android mobile OS has caught consumer imagination and very few users will not have seen the ensuing press following this litigation. It has been brought well and truly into the public eye.

The nature of patents in software and number of patentable pieces of code in any smart phone device, including Samsung and HTC, means that it is very easy for patent owners to sue and be counter sued. The extent of this war of litigation has led many people to ask me who can win from all these counter suits but the lawyers charging tens of millions of dollars to litigate these suits. Frankly, I am not sure that I can disagree with this analysis although some of the patent holders who are trying to charge a royalty per device could become winners.

In short the patent litigation creates a risk to open source as it may carry two consequences. Firstly if the litigation was

successful the patent holders might be in a position to charge a royalty that in effect could create a premium on the cost of open source usage. Secondly it could enforce a barrier to entry, where users are put off the usage of open source due to fear of litigation. However, my personal view is that, this will not ultimately be successful. The continued success of the Android phones in the smartphone market place, sharply followed by Mozilla and Canonical's mobile OSs may be a good indicator of the ongoing force of open source that is moving in this direction. The existence of the litigation itself could arguably be considered to be nothing more than evidence of the success of open source's commercialisation in this way and the ongoing fear being caused in the established market places in which it is competing.

Android is clearly on of the most disruptive forces to create waves in a market place. However, I am seeing ongoing waves of disruption that may not be a tsunami, but over time may build up to one. These involve the creativity of multiple organisations seeking forms of funding not only to be profitable but going back to Stallman's thoughts on decision making, where actions are motivated by various reasons including the individuals' values. For some individuals and organisations that would be generating profits and meeting share holder value, but for others in the open source space, it may be as simple as paying for their own excellence, i.e. being able to generate enough revenue for sustainability.

Disruptiveness and that approach thinking outside of the box of convention may be much wider than looking to software for profitability. Emerge Open is a UK company that was formed was to find sustainable means of funding for Open Source projects. Despite accepting the current necessity for donations the founders are working on getting models to pay for their sustainability off the ground. IT Recruitment is a market worth tens if not hundreds

of billions globally. As one of Emerge Open's directors is from a recruitment agency background, they are working on a model where recruitment agent services will be provided and function just like a normal recruitment agency but all the profits will go to funding Open Source projects. In doing so they are using the considerable pool of open source talent to specialise in a space that is well known to them, but where they offer customers the added value that the profits made by their business will go back into the market sector, i.e. open source.

App stores have become a massive part of software distribution. Apple and other smart phone companies have created a very profitable, although potentially transient market place for their profit in distributing native apps (apps designed or modified to work on their devices, develop or modified through their APIs and for sale or distribution from their stores). The reason that this may be a potentially transient model is the improving functionality of HTML5 and other coding tools which make Web Apps better. As Web App functionality improves, the lives of developers will become easier with Apps being able to work across devices without the need for tailored development. Whether Web or native is the final route, running an App store may be a great source of profit. Contributing Apps for distribution through those stores may also be a source of profit. There are some issues however with respect to the infrastructure of App stores including the terms and conditions being put in place.

In the early analysis we worked through the issues which copyleft may raise and of course the fact that there may not be additional terms and conditions added to the code's licence if it is a GPL one. This could easily not be an issue but unfortunately some App stores have not simply ring-fenced the code and its licence terms with the store's terms being restricted to terms of usage and sale. Where App stores have not taken adequate care to

ensure that their terms do not add extra conditions to the licence or have otherwise not complied with licence requirements e.g. providing access to the source code or allowing modifications to be licensed back on the same terms, the licence may be infringed by the app store's distribution. The theory would be that this is not an issue as the developer who owns the app must have put it onto the store for distribution under the licence, knowing the terms that the store imposed and would not do so if they were unhappy with it. If the store's terms changed over time the developer could always remove the app. Some may take the view that an unenforced copyleft licence is no different from a permissive licence, but there is no obligation on a developer to enforce their licence copyleft of otherwise.

The collaborative nature of open source can however be problematic in this situation. If the app is being distributed in a potentially infringing way, then the collaborative nature of development and the fact that it is unlikely that one individual created a piece of code, and even if they appear to then others code is likely to interface with it, may prove problematic. Apple felt the wrath of the FSF who own the copyright in some of the VLC media player's code (it having been assigned to them by developers under their contribution agreement). However, their response to the FSF may seem to many to have backfired from an open source perspective. Rather than fight the issue, they simply removed the GPL code from their app store in response to the threatened litigation. They were able to do so by relying on the safe harbour provisions of the US copyright legislation under which they are merely a publisher. This reaction appears to be perfectly legal but did not enhance the reputation of open source, nor fix the potential conflict between the Apple app store and GPL.

There are undoubtedly many, many ways to commercialise open source software. Some are more obvious than others. Some are more problematic than others. The question of course will be how successful will they be and whether they can sit with the ideological stance taken by the early developers of free software, to allow for funding the long term sustainability of FOSS. At this stage in time, that remains to be seen. What is clear however is that the commercialisation of FOSS comes with technical, legal and ideological issues which will certainly impact its success.

There is no such thing as a free lunch. Is there such a thing as a free piece of code?

Amanda Brock advises a diverse group of companies on commercial and technology law issues from her London base. Having been General Counsel, Canonical – lead commercial sponsor of the open source operating system Ubuntu, for 5 years, where she managed the worldwide legal function she has particular expertise in open source software, cloud computing, big data and device manufacture and distribution.

With over 15 years experience of commercial and IT law, gained working as an in house lawyer, including roles as European Manager at DSG International where she was the first lawyer at the ISP, Freeserve (UK's first and biggest .com IPO), UK Legal Director, Aramark and General Counsel Nicole Farhi & French Connection, she has diverse experience which lends to her very commercial approach to legal advice.

Amanda is as a solicitor , admitted in England and Scotland and has a Masters in IP and IT law from Queen Mary and a Masters in Comparative Jurisprudence from NYU. Amanda is a member of the Advisory Board of the QM, University of London Open Source Centre of Excellence and has spoken internationally and written extensively on Tech and commercial law. She is one of the founders of the QM Legal Incubator currently being set up to provide legal advice to the UK start up market place. Her clients range from start ups to Mozilla.