

## TRANSCRIPT OF SCL FREE/OPEN SOURCE SOFTWARE ('FOSS') TALK, 28 MAY 2009

**RICHARD KEMP, KEMP LITTLE LLP**



In September 2008, Gartner published the results of their user survey of Open Source Worldwide. They surveyed 300 companies and found that 85% of them were using FOSS then (the survey was conducted in August and September of last year) and that the remaining 15% all had plans to be using FOSS within the year so, by around now, all of the survey sample would be using FOSS. Out of that sample, it was estimated that about 25% of all the software they were using was FOSS. So a large part of the code base big global corporate companies are working with is FOSS. But of those 300 companies, about 70% didn't at the time of the survey have a corporate policy framework in place to support their use of FOSS.



- Agenda
  - Setting the context:
    - CIO's boon and GC's burden
  - Issues with FOSS licences
  - Why it matters – current cases
  - Handling FOSS safely



See also the KL Introduction to Open Source at:

[http://www.kemplittle.com/PDFs/Article\\_IntroductionToOpenSource.pdf](http://www.kemplittle.com/PDFs/Article_IntroductionToOpenSource.pdf)



**Free/Open Source Software ('FOSS').**

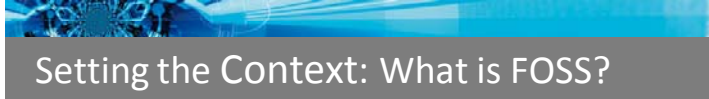
Presentation by Richard Kemp, Kemp Little LLP, 28 May 2009.

These materials were produced for a talk at Society for Computers and Law London meeting and do not constitute legal advice.



The Gartner survey was conducted before the onset of the financial crisis so those trends towards increasing FOSS use are likely only to have quickened over the last six to eight months as cost reduction for most businesses has become an imperative; and many people see FOSS as a way of reducing the cost base for software in a particular organisation. So in a sense, that's the context for the first part of this evening's discussion, which is around where in the organisation the benefit and the burden of FOSS lie. So I'll be saying a little bit around:

- what FOSS is;
- who gets the benefit;
- who gets the burden;
- a number of common issues with FOSS licences generally;
- why it matters - having a look in this area where there is not an abundance of case law; what the settlements and the cases there are actually say, and
- going on and having a look a little bit more about how FOSS can be handled more safely in the organisation.

### **What is FOSS?**



- A licence granting certain freedoms
- A range of associated licensing techniques
- Many different types
  - Clarity
  - Length
  - Legal effect
- Mainstream



The first thing about FOSS is that it's no different from any other type of software: what's inside the tin is the same - whether it's proprietary software or FOSS. What's different is the label on the tin, and the best way of seeing FOSS is as a range of associated licensing techniques: the FOSS licence grants the user certain freedoms and it is in those freedoms that the licensing regime for FOSS differs from the licensing regime from proprietary software.

There are hundreds and hundreds of different types of FOSS licences. The Open Source Initiative (OSI) itself has approved more than 70 different types of licence and they vary in clarity. Some of them have been legally drafted, they can run to 4 or 5 pages, some of them have obviously not been drafted by a lawyer and are more or less clear. We have seen some that literally run to a single sentence. So the legal interpretational skills that one brings to bear to FOSS are different and that is

almost the first thing to say: whether you're inside the company or outside the company, working with FOSS is different. In a sense, you've got to throw away your text books - Chitty and Copinger - and really adopt a different perspective for working with FOSS and interpreting its legal effect. That's one of the excitements and challenges about advising on FOSS today.

Not many of these licences have yet been litigated so there's not a lot of settled case law right now. You can't go to the books. The way that the disputes tend to surface is rather different. You tend not to get a letter before action which is the first thing that you hear about it. You need to think of the development world of the FOSS community as its own kind of ecosystem and disputes tend to arise by a swelling of murmurings of discontent within that community which are taken up at a progressively higher level by one or more of the FOSS bodies, typically the Free Software Foundation (the FSF) in the USA or the FSF in Europe (the FSFE).

So whilst the code isn't different, it's the licensing techniques that are different and particularly the interpretational skills that you apply to those licensing techniques.

### **Who gets the benefit of FOSS?**



The CIO's boon...

- Reduces time to market for new products
- Lower costs
  - Licensing
  - Maintenance
- Resources saved for differentiating applications
- Cool




Now, I mentioned before that FOSS is now in the mainstream. It's increasingly widespread in the organisation. The highest adoption rates for FOSS are in the operating system area - Linux is the classic example - and also in middleware and infrastructure applications where commonly used FOSS products are My SQL and Postgres in the database world; Apache in the applications server world; and Perl and PHP in the development tools area. At this level, FOSS is not only a maturing but is a mature market place.

But FOSS is increasingly being used in the applications world, it's much more widely deployed now in the applications that companies use and Gartner's survey found that such use was increasing to the point where it is now in excess of 20% in each of the areas of content, CRM (customer relationship management); supply chain management; and enterprise resource platform management.

So it is definitely the CIO's boon. It can reduce the time to market for new products so the CIO can put his expensive in-house software development resource to work on higher value added work for the company and use FOSS to deal with the commoditised areas; it reduces dependence on suppliers - use of FOSS is seen as one of the ways of diminishing your reliance on Microsoft or Oracle or the big platform providers; and it can lead to lower licensing costs for licensing and maintenance.



Basically, you are saving your resources and deploying them on the applications that differentiate your business and confer competitive advantage. There is also a certain coolness factor as well in using FOSS – the development world likes to work with FOSS. So there are lots of reasons why your CIO will like using FOSS, but by the same token it is the legal department's burden.

### Who gets the burden?



...and the GC's burden

- Inbound transactions
  - Procurement
  - Internal development
  - M&A
- Outbound transactions
  - Licensing applications and products
- Scope of governance
  - Developers to sales team



Up to now, you could say that the responsibility and the benefits of FOSS have been misaligned. The responsibilities tended to be in the legal department whilst the benefit has tended to be in the CIO's group. You see that in a number of different types of areas and the best analysis is to look at FOSS in use in:

- in-bound transactions;
- in the development area in the company itself; and
- on out-bound transactions.

**Inbound transactions.** On procurement, the classic horror story is a company will go outside, maybe offshore, for software to be developed. The developer gets under time pressure; maybe it takes longer than he thought to write the software, so in order to meet the timelines, he bakes in an FOSS component that the company that is procuring the software doesn't know about it, links its own proprietary software to the say GPL software that the developer has been developing in the wrong way and, before you know it, when the procuring company is licensing out that software to its customer, it is finding that its own code is being GPL'd by the copyleft provisions which I will come on to in a moment.

That's the classic horror story. We haven't actually seen one of those happen yet, at least in reported cases. What we have seen in the M&A world is where at the due diligence phase, most targets will say, we don't use FOSS in our software code base and then the acquirer will run the code base through Black Duck or Palameda or one of the other code indicator tools and invariably in our experience FOSS will turn up.

I was on a platform with a FOSS expert from a leading global software developer a couple of years ago and he said in the previous year they'd had a number in the mid teens of software companies that they'd acquired that year and, all but a small handful of them said they didn't use FOSS and when they ran the code indicator tool, they found FOSS in all but a very few out of those deals. That gives you an example of how widespread FOSS is in use and also how there is a communications logjam between the development group and management.

**Internal development.** In the internal development area, the classic case is of the internal software developer pulling off GPL code off Source Forge, or one of the places on the net where FOSS is available, and using it maybe contrary to corporate policy, maybe not because as we have seen most of these companies don't actually have these policies in place and again causing problems with the company's code base.

**Outbound.** Increasingly, these days where we come across it in our work at Kemp Little, is where a customer taking a software application will want to make sure that there are appropriate contractual terms protecting it, the customer, from use of FOSS, so in licensing applications and products, there is an increasing focus on not only generic warranties and liabilities and indemnity provisions, but also on specific FOSS product issues themselves.

**A number of common issues with FOSS licences generally**

| Basic FOSS Freedoms   |  |  |
|---|--|--|
| <p><b><u>FSF/GNU GPL</u></b></p> <p>Four Freedoms:</p> <ol style="list-style-type: none"> <li>1. to run program for any purpose</li> <li>2. to view and modify the source</li> <li>3. to release mods to the public</li> <li>4. to publicly redistribute source &amp; object</li> </ol> <p>■ 'copyleft'</p> | <p><b><u>OSI OSD</u></b></p> <ol style="list-style-type: none"> <li>1. free redistribution</li> <li>2. source code availability</li> <li>3. derived works</li> <li>4. integrity of Author's source code ('patches')</li> <li>5/6. no discrimination against people or use</li> <li>7. no extra licences for redistributed software</li> <li>8. licence must not be specific to a product</li> <li>9. licence must not restrict other software</li> <li>10. licence to be technology neutral</li> </ol> | <p><b><u>Closed Source Compatible</u></b></p> <p>Generally:</p> <ul style="list-style-type: none"> <li>■ copyleft rules out closed source</li> <li>■ not required to disclose source mods</li> <li>■ allows commercial use &amp; redistribution</li> </ul> |

**Types of FOSS licences.** So the CIO's boon, the GC's burden, is the rule of thumb historically speaking. So what are we talking about in legal terms when we are looking at the FOSS freedoms? The best thing is to think of FOSS licences as a spectrum, with more permissive at the one end and less permissive on the other end and on this side, the less permissive ones are the GNU GPL, the

General Product Licence, which was published by the Free Software Foundation. This is the kind of evangelical wing of the party if you like. These are the guys that are more antithetical to the proprietary software world. They trace their roots right back to the counter culture of the '60s and there is still a little bit of that approach in the FSF's view of the world characterised by Richard Stallman, the FSF's leading light.

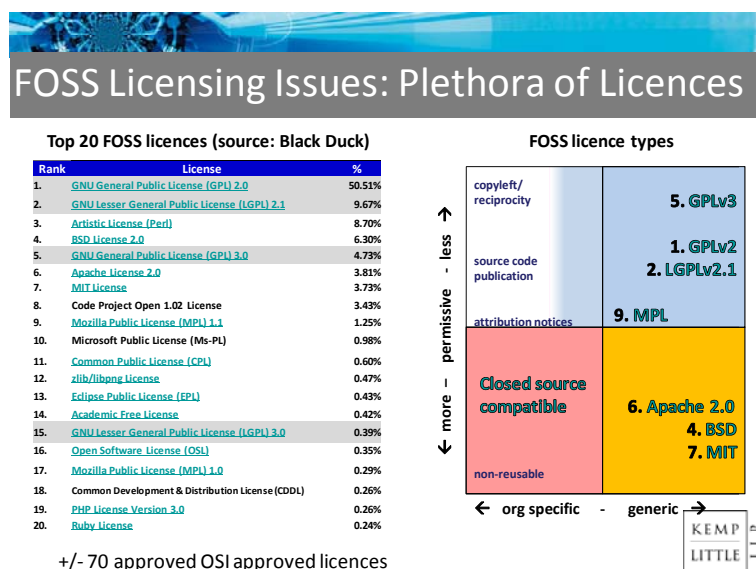
There are four basic freedoms that the FSF espouses:

- that a user can run the programme for any purpose;
- that it can view the source code and modify it;
- that it can release those modifications to the public; and
- that it can publicly re-distribute both the object code and the source code and that links in to this copy left issue which I will come on to in a couple of slide's time.

The FSF is pretty evangelical about these four freedoms. The more pragmatic wing of the party if you like, is the Open Source Initiative (the OSI) who have come up with this Open Source Definition (OSD) which is a list of 10 criteria that FOSS software must meet for it to meet the OSI's definitional requirements. I won't go through them all. They have approved about 70 licences as meeting these requirements. They start off with requirements for free distribution, free source code availability and freedom to make derivative works, so to that extent, they are akin to the GPL world. Then you tend to get these proportionality, non-discrimination, and technology neutrality requirements and these are very much the more watered down version of the copyleft principle that the FSF espouses.

Then on the right wing of the party if you like, you have closed source compatible. 'Copyleft' rules out closed source compatible. Using closed source compatible code you may not be required to disclose source code modification but commercial use and re-distribution are generally allowed.

**Plethora of FOSS licences.** This slide puts some flesh on the bones. The first licensing issue that anyone who is advising in this space will come across is just how many licences there are out there.



I took the table on the left hand side from Black Duck, which is one of the leading providers of code indicator tools, and this is a ranking of the top 20 FOSS licences. The first thing to notice is just how prevalent the GNU GPL is, which is the licence base for Linux, the most popularly used FOSS operating system which accounts for about half the FOSS world. If you put the GPL, and its cousin the lesser GPL, in versions 2 - which are at numbers 1 and 2 in the chart, and versions 3 - which came out in 2007 - along together, you are getting on for two-thirds of the total FOSS licence world being likely to be licensed under this more evangelical wing of the party's licence, the GPL and its cousin the lesser GPL.

Next, you have the academic licences - the Berkeley Software Distribution (or BSD) licences and the MIT licence - also in the top 10. These are a lot more liberal, they are a lot more permissive. We come across Apache a lot in our work for particular clients. Again, a feature here is that the top 10 licences will probably cover about 95% and the top 4 probably account for 80%.


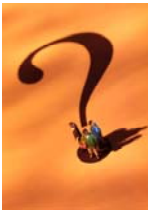
The graph on the right borrows from the preceding slide where more permissive is down at the bottom and less permissive is up at the top, plotting that against the specificity of the licence. So in the top right hand corner in blue are the least permissive, most generic licences - the GPL and the LGPL licences; in the bottom left hand corner are the closed source compatible; then you have Apache, BSD and MIT in the bottom right corner - very generic but quite permissive; and in the middle you have Mozilla, the MPL where the axes meet.

So the first issue is there are a heck of a lot of licences out there. Twenty here and as I mentioned the OSI itself has approved 70 and that probably goes up by 10 - 20% every year.

**Copyleft.** The second issue is copyleft. I won't say too much about this. It's probably the area that has exercised most academic legal attention but the real point is this interpretational one of not getting too hung up with your typical legal forensic tools.

## FOSS Licence Issues: Copyleft

- GPL v2
- Act of running Program not restricted
- Derivative works – Section 2
  - 2b: Any work that “contains or is derived from the Program” must be licensed under GPL
  - Does linking GPL licensed code to your code create a work “containing or derived from” GPL code?
    - Different types of linking
  - Consequences of getting it wrong
  - Impediment to FOSS uptake?
  - Need for better guidance?



On copyleft, as I mentioned, GPL version 2 states that the act of running the programme itself isn't restricted. The fun starts with Article 2b which says that any work which is your company's work (let's call that **A**) that 'contains' or is 'derived from' 'the program' – i.e the GPL code (let's call that

**B**). must be licensed under the GPL. And as we've seen, the GPL has these four freedoms: you can't stop derived works being made; source code has to be made available, and so forth. So if your client with your product **A** were hoping to keep your proprietary licensing terms quite tight, and that work 'contains' or is 'derived from' GPL licensed code **B** then you will be licensing your software **A** under different terms from which you thought you were licensing it under.

The central question there is: does just linking code **B**, the GPL licence code, to your code, code **A**, create a work that contains or is derived from the GPL code. And this is where it can get quite technical if you do ever need to go down to it in that degree of detail.

There's a lot of debate about what types of link can be sufficient to cause this GPL Article 2b 'containing' or 'deriving from' to take place. There are lots of different ways in which **A**, your code, can link with or to **B**, the GPL code, particularly where the GPL code is Linux. Links can be static - 'once for all linking; dynamic linking - where the link is made at boot time; plug-ins; remote procedure calls; and templates and interpretative languages for example. Equally, **A**, your code, can inter-operate with code **B** through an application programming interface, an API, where on one view there is no linking taking place at all. So a whole host of ways in which products can be linked and lots of different interpretations as to whether they amount to 'containing' or 'deriving' in the GPL sense.

There is no settled case law on this yet. The generally accepted view is that article 2b works. As a matter of contract law or copyright law, the copyleft construct is something that holds water and that is probably the key take-away. You can't dismiss it as not working at all. The extent to which it works and all the other issues are probably secondary. The Free Software Foundation has published extensive FAQs to GPL's versions 2 and 3 and they go into a certain amount of detail about the degree of information communicated, the closeness of the links, the closeness of that communication, but as an interpretation tool, the FAQs are probably a little bit self-serving as evangelising the FSF's view of the world. This word "contains" in Article 2(b) is a little bit mischievous because it goes beyond normal copyright law. But in general, yes, it probably works. As to the precise extent, no settled case law yet.

But the consequences in getting it wrong can be quite severe. You may find that code **A** that you thought you could control the licensing of, and that you wouldn't for a month of Sundays want to give out the source code for, is now actually required to be source code licensed to anyone who wants it. Or, if you don't want to do that, you may find that you can't use **B**, the GPL code, with **A**, your code in that way. Or, you might have to write around the GPL code to find another way of doing it at a time when presumably you are launching a new product and you are very strapped for time.

The consequences of getting it wrong are such that even the FSF is starting to realise that this is an impediment to the take-up of GPL FOSS. There is a recognition that there is a need for better guidance, that people need clearer rules of the road, that this slightly mischievous language that they have put into Article 2b probably hinders more than it helps. So that is why the debate is moving away from the over-reaching nature of Article 2b. to try to get a little bit more clarity, not for the sake of the legal principles, but to remove this impediment for the take-up of FOSS. So say two years ago the detailed copyleft analysis was probably the number one issue. It seems to be moving back from that now on the basis that well, it probably works.

**Patents.** The third issue that I wanted to have a quick look at was patents. Again, there was early on a recognition in the FOSS community that the FOSS concept could be wrecked if patents were invoked to stop FOSS being publicly made available. That's why if you look at GPL version 3, which came into effect in June 2007, section 11 is effectively an obligation from the licensor not to derogate from the grant in respect of patents: that you can't give with the one hand with copyright all these good things, and then try and take them away with the other hand when it comes to patents.

## FOSS Licence Issues: Patents

- Early recognition in the FOSS community that invoking patents could wreck FOSS objective
- Approach of many FOSS licence is to require essential patent rights to be licensed
- Cross-licensing patent deals
- GPL v3 – 29 June 2007 - Section 11
  - Non-derogation from grant obligation for patents from all program contributors



It is again quite an esoteric area when you get down to the detail. The technical thing on patents is to whether the use of FOSS would amount to an infringement of a patent: is the FOSS use at a level of abstraction different from that to which the patent relates? That hasn't actually been tested yet.

So those were three specific particular licensing issues that we see coming across our bows. It is the plethora of licences, the fact that there are so many of them and how do you interpret them. The second is the copyleft issue and the third is patent licensing.

### Why FOSS matters

## Why It Matters – the Cases: Germany

- Cases brought by gpl-violations.org
- 4 judgments
  - Sitecom (19.5.04, Munich)
  - Versatel (21.2.06, Berlin)
  - D-Link (6.9.06, Frankfurt)
  - Skype (12.7.07, Munich)
- Claims for:
  - Cease licence/© infringement
  - Account/information
  - Damages and costs
- No court actions in last 12 months
  - Success of approach to settlements
  - Grace period, payments, negotiation, reporting



Now, why does all this matter? Why are we here? Why is FOSS rising at the agenda? Why are these legal issues topical? Two examples. The first is the German experience where there is a very active organisation set up by a guy called Harald Welter called GPL-violations.org who has really picked up the ball and run with it in Germany to the extent that there are now four judgements, Seitcom, Versatel, D-Link and Skype, all in the three year period between mid 2004 and mid 2007 where the German courts recognised that Harold Welter's organisation had locus to bring the claims. These claims were typically for contract, licence and/or copyright infringement; and also for an account of information - GPL-violations.org wanted to know where the software came from, who had been involved in its dissemination and distribution; and then also for damages and costs.

Those cases were pretty successful from the FOSS community's point of view to the extent that there haven't been any new court actions in Germany over the last 12 months and they now have quite a well worn path, a well worn approach, to settlement negotiations. They will give a grace period, payments are made, the settlements are negotiated, there are reporting obligations put in place. Germany is quite a good case study for the FOSS approach. They don't really want damages, they just want to make sure their software gets out there and their degree of success has been such between mid '04 and mid '07, that they can now do this routinely.

### Why It Matters – the Cases: Cisco/FSF (20.5.09)

- CyberTAN used GPL/LGPL code to customise Broadcom standard Linux distribution
- Broadcome embedded code in one of its chipsets
- Linksys adopted chipset in its wireless broadband router WRT54G
- Cisco bought Linksys for \$500m in 2003
- 11.12.08: Free Software Foundation sued Cisco of GPL violation
- 20.05.09: Dispute settled - key terms:
  - Cisco appoints a Free Software Director for Linksys
  - C/LFSD makes ongoing compliance reports to FSF
  - Cisco to notify Linksys customers of their rights under GPL/LGPL
  - Cisco to publish licensing notices
  - Cisco to make 'complete and corresponding' source code freely available
  - Cisco to make a monetary contribution to the FSF



The second example which I quote largely because it is topical, is the Free Software Foundation's announcement on 20 May 2009, so just over a week ago, of the settlement of its long running dispute with Cisco. Very briefly, this centres on Broadcom, the 'system on a chip' maker based in Southern California, which around 2003 used a company called CyberTAN to customise its standard Linux distribution. The 'Linux distribution' is the Linux code and its accessories and the various FOSS components that you use with it. So Broadcom got CyberTAN to customise its Linux distribution and it used GPL code to do that obviously. Broadcom embedded that code into one of its chip sets and Linksys adopted that chip set in its wireless broadband router. About five years ago Linksys was bought by Cisco for US\$500m.

The period between 2003 and 2008 is an examples of the murmurings of discontent from the FOSS eco community I mentioned earlier getting louder and louder and the FSF took up the cudgels on behalf of the FOSS community in 04/05 and then tried to get compliance with its licence from Cisco,

didn't, and ended up suing Cisco at the end of last year. The dispute was settled last week. The key terms, and these are quite interesting again from the point of view of governance and compliance, are that Cisco agreed to appoint a Free Software Director for Linksys; the Free Software Director is required to make ongoing compliance reports to the FSF - that's the future of FOSS: a compliance officer and ongoing reports if you are caught; Cisco has got to notify the Linksys customers of their rights under the GPL; it has to publish the licensing notices, it has got to make complete and corresponding and up to date source code available on its website; and it had to make a monetary contribution to the FSF. That is a classic example if the FSF gets hold of a case, decides that it has got legs, these are the terms that it will settle on, compliance person reports specific and wide publication and notice obligations.

### **Handling FOSS safely**

Last, we come on to what do you need to do in your organisation or your client's organisation to handle FOSS safely. There are really two things here - the contractual matrix and the governance matrix.



## Handling FOSS Safely: Contractual

- Inbound transactions
  - M&A
  - Development and Licensing Inwards
  - Procurement – RFI/Ps, tenders, etc
- The in-house development community
  - See governance
- Outbound transactions
  - Customer services and licensing agreements
- Warranties, liabilities and indemnities



We have seen in the contractual area that we have got this analytical framework, in bound transactions, the in-house development community which is really where the governance rule applies and outbound transactions.

On inbound M&A, you need to build a machine that can see your target company's code base, can make sure that there is no unexpected FOSS in there, and, if there is, build in sufficient time before closing when the target can remediate that code. When you are buying in software or having it developed externally and it goes through your procurement machine, we are seeing increasingly that companies are not relying on warranties and liabilities and indemnities in the general sense, but having specific parts of their procurement process that apply to FOSS, so there may be a specific obligation to let the procuring company put your code through a Black Duck or Palamedia code indicator tool, for the supplier to bear the cost of that if it shows undisclosed FOSS, and for there to be specific remediation obligations.

On outbound transactions, the flip side of the coin, customers are increasingly concerned to see specific FOSS obligations in there. This is important because if you think about the types of FOSS obligations, very often they are not caught by generic warranties of no infringement, the liability provisions may be different, you don't want monetary damages, you just want the code remediated, say, and infringement and claim indemnities might not apply in the ordinary way.

So the final point, and this is really what we are seeing happening today, where the CIO's boon and the GC's burden are coming back into balance, is through the governance mechanism. If you go back to that Gartner survey that I mentioned at the start, then seven out of ten of these big global customers that Gartner surveyed didn't have compliance policies in place. That is starting to rise quite rapidly and it is the way from the organisational point of view of the legal department ensuring that the people that get the benefit also get the responsibility for managing the risk.



## Handling FOSS Safely: Governance

- The **policy** context – towards best practice
  - A written open source policy: vision – strategy - policy
- The **people** context
  - FOSS compliance officer – project team – working party
  - Involve all stakeholders (development, legal, marketing, leadership)
- The **technical** context
  - Review – assess remediate
  - Code indicators – Black Duck, Palamida, Fossology
  - Code audits
- The **process** context
  - Awareness training
  - Policy communication/buy-in
  - Rules of the roads – key 'do's and don'ts'?
  - Approvals and exceptions
  - Reusing FOSS code and code registers



We are seeing a much more driven governance framework. Briefly, I'll look at this from four contexts.

First, the policy context, which is moving towards best practice, where people tend to have a vision statement, then a strategy and then the policies and procedures, and this is I think is something that Pete will be coming on to much more in his presentation.

Secondly, the people context - this goes back to the Cisco case where you tend to get a compliance person, someone charged with the responsibility within the organisation, maybe a project team and a working party looking at FOSS use across the organisation and the involvement of all the stakeholder groups; these will be classically the development team, marketing and business leadership, in addition to legal.

Thirdly, the technical context: review, assess, remediate - your classic mantra for assessment projects like this. You will review what you are using, you may put it through a code indicator tool, you will assess what you need to do and will take remedial action if necessary. We are seeing Black Duck particularly, but also Palamedia and FOSSology as increasingly used in many large organisations. At the end of the process you might repeatedly, say at annual intervals, have code audits taking place.

Fourthly, we get involved in the process context a lot in awareness training, what to look out for. There is a lot of work around communications internally to ensure buy-in from all the stakeholder groups, the rules of the road, like all of us the development community may not individually have a very long attention span, they need a page of dos and don'ts otherwise it is not going to sink in. What are the approvals mechanisms, what are the exception mechanisms within the organisation and then do you keep registers so that FOSS code, say a Java applet that has been used for one part of the business and cleared, that that can be used within the organisation.

So really this brings it full circle. The governance regime is how you align the boon and the burden; it's the way to make sure that rather than the legal department responding reactively to risks at the eleventh hour, you can put in place in the development community, perhaps under the CIO's guidance, these governance rules that align the responsibility for using FOSS safely with the benefits it brings.

Thanks for listening and I'll now hand over to Pete.

**Free/Open Source Software ('FOSS').**

Presentation by Richard Kemp, Kemp Little LLP, 28 May 2009.

These materials were produced for a talk at Society for Computers and Law London meeting and do not constitute legal advice.