CS2406 Open Source Basic Viva Questions

1. What are Gnome and KDE, and are they compatible?

Gnome and KDE are two large desktop environments. Both sit on top of the X server on Linux and work with a window manager and framework services. The framework service provides everything from color and font preferences to communication between other application services running on that desktop. Although Gnome applications can run in KDE with the installation of the kde core libraries and vice-versa. The two are compatible on an X level primarily. Gnome uses GTK+ for its GUI presentation while KDE uses Qt. Ironically Novell owns both technologies yet has not published a roadmap or any clue if full interoperability is on the cards.

2. What open source mechanisms exist to get software updates?

Several mechanisms exist today. For delivery of rpm based packages you can use yum or red carpet. For debian packages apt-get, for gentoo, ermerge and for Fire fox there is an xpi installer. Some users will only use source in which a cvs update is all that is required.

3. What are svn, bit keeper and cvs. If you have used any of these tools what enhancements would you like to see if any?

svn, bitkeeper and cvs are commonly used revision control systems in the open source community. For all the advantages that these tools bring, like networked based code commits and ease of editing there are a number of features that the professional complementary tools provide. One such feature is a change set, the ability to edit a group of files in your own working copy and manage that list of changes locally. Merges and conflicts also vary between tools and how revisions to a file are tracked can also have an impact on tracking deltas in source code. If you are happy with the tools as is then that is a good answer too.

4. What is the latest version of the Linux kernel, and what are the main improvements? A3.

The latest stable version of the linux kernel is 2.6. Most of the main distributions support this version, the odd-number based kernel releases used to represent a kernel in development mode. The latest update version of the kernel is now at revision 2.6.17. Although developers would be normally looking for a 2.7 kernel to appear for development work the 2.6 kernel is still where most of the new development is being carried out.

5. You've deployed your open source application, and it stopped working. What tools can you use to debug open source software?

Most of the popular open source projects use a logging infrastructure to log to a application log file or to the system log files. First look for any unusual messages there before bumping the log level up if you can. Use platform specific tools if possible, like the debug interfaces in net beans and eclipse can generate very verbose Java stack traces allowing you to pinpoint a line of code. If not you may have to resort to the gnu command line debugger gdb. If the application has already crashed, or you have run the command gcore on a running process you will be able to point gdb at that core file. gdb will be able to show you the stack trace of the last running thread and often the crash will be the result of accessing an invalid memory address like 0 (or null). For kernel level issues there is a kernel debug tool called kdb although unless you are working on the kernel most companies wouldn't necessarily expect that level of experience.

6. What open source components have you used?

Although this should be an easy question, don't get caught trying to guess a version of Linux or Apache. The current corporate versions of Linux used are Redhat Enterprise Linux 4.0 and SUSE Linux Enterprise Server 10. Knowledge of the personal editions of those distributions, like SUSE (10.1) or Fedora Core 5 or earlier should be more than sufficient. The enterprise editions primarily have kernels tuned for large scale systems and additional services such as clustering deployment and networking booting. If you used Redhat Enterprise edition don't forget they only shipped RHAS 2.1, (before the enterprise name was used) RHEL 3 and now 4. For the other popular open source components, the latest stable version of Apache is 2.0.59. JBoss is at 4.0.4 and MySQL is at 5.0.24. Many corporate employees will probably be at least one version behind the latest developer release so describing how the latest version works or doesn't work with earlier versions may earn you extra credits.

7. You've been asked to shortlist two open source components for your companies application. How can you tell the difference between a stable relatively bug free project and a new risky alternative?

Apart from obviously downloading each and trying them out and seeing how long each takes to configure there are other clues about how active and stable each project is. Some examples would be to look at the project status on sourceforge.net or the home site for the project. Check the forum and bug postings if they have them, are bugs being fixed, forum questions being answered? Next look for the rate of change, you may not be able to keep up with a project with a rapid rate of change and need to debug why your application stopped working when you got the latest snapshot. Finally, do they release milestone or snapshot binaries, if not consider the project under constant or stopped development.

8. What web service based technologies are available for open source developers?

The original web service technology, XML over RPC is used by many open source applications; most web service support comes in the form of language specific binding tools such as SOAP for PHP, JAX-RPC or Axis for Java, SOAPpy for Python. However several services do expose the REST service method which uses regular get and post http calls and is often easier and quicker to use.

9. What are the risks and trade-offs between using static versus dynamic libraries with open source software?

Dynamic and static libraries are primarily used for C and C++ based applications and designed to provide different levels of compile and runtime binding. Building with a static library means that your application included all the relevant code from the open source project you were using. This should mean that your application should run fine our of the box, however the downside is that you may forever be linked to a bug in the library code you used so that if a fix is needed in that library you need to rebuild your application too. Static linking is also treated differently that dynamic linking by some licenses. The upside to dynamic linking is that by separating the application to call the library code at runtime your own application is going to be smaller, you will also benefit from bug fixes made in that library. The downside is that the application binary interface (ABI) may change which means in rare cases you will have to release a new version of your product anyway.

10. You've been asked to bundle MySQL in your companies new product. The license of the final product has yet to be decided but you read somewhere that MySQL is GPL. What would you do next?

If this is for a commercial vendor and you don't know your companies licensing model you should defer this question to your internal legal representative or equivalent. There are many conditions applying to software like MySQL that is dual licensed and what is allowable on your own open source project may require a different license for your own company.

11. What is Free and Open Source Software?

Generally, the name describes software that is licensed with fewer restrictions than proprietary licensing models, such as "per copy", "per use" object code only licenses. The term "free software" often refers to software that is licensed under the General Public License ("the GPL"). "Free" does not refer to cost, as the GPL does not preclude charging for distribution of licensed software, but rather it refers to the lack of constraints on using the software. However, to prevent intermediates from imposing their own constraints, the GPL includes provisions precluding the addition of constraints.

The term "open source" often refers to free software as well as software licensed under other licenses generally considered to be open source licenses. Open source licenses might include provisions regarding limits on constraints, attribution requirements, no-warranty notices and other important provisions. Open source software often has fewer constraints on intermediate parties, but possibly more constraints on downstream parties (because an intermediate party might be free to add constraints to the software as they distribute it).

The question of whether or not a particular license is to be considered an open source license has been widely debated. The Open Source Initiative formulated a set of tests that they propose as a definition of an open source license).

12. Linux is perhaps the most widely known example of open source software. What are some others that people may not be aware of?

The Apple Macintosh's underlying operating system is open source software. Other examples include the Apache Web Server programs used by many of the web servers running today. The GNU operating system, including its many programming tools, development environments and programs, is also free software.

13. Typically, open source software user does not sign a license agreement and the software does not have implied consent via shrinkwrap or clickwrap notices, so how can an open source software user be subject to the license?

Software is copyrightable, so most uses, copying, modification and distribution of copyrighted software requires some license from the copyright holder, otherwise the user/copier/etc. would be a copyright infringer, unless a defense such as "fair use" applies. A license agreement might grant a license in exchange for licensee consideration, in which case assent of the licensee would be required to form a binding agreement. However, if a license does not impose any new obligations but simply releases obligations imposed by copyright law, a putative licensee would have no reason to reject the license.

14. What are the main legal issues that can arise with free and open source software licensing?

The first issue is who the copyright holder is and the second issue is what license I might have to use, copy, modify or distribute the software. Then, the next issue might be what is allowed under the licenses I might have. Where the copyright holder is known and amenable to additional non-exclusive licenses under different terms, such licenses might be negotiated. For example, where a company wants to incorporate particular software into their product and distribute that product on a proprietary basis (e.g., per-copy licensing, a prohibition on reverse engineering, etc.), the company might sign a license agreement with the copyright holder that grants the company more rights to be proprietary than under the available open source license.

When a company decides to release its own software under an open source license, there are issues of which license to use, and whether to create a new license.

Large companies also need to have some process for tracking and evaluating open source licenses that might apply to software their employees are using. Since much open source software can be obtained at no cost, tracking software licenses in a purchasing department will miss most open source software.

15. Can you give some examples of actual issues you've worked on that deal with free and open source software?

We've advised a number of clients on the scope of various open source licenses when they looked to see whether they would have a license to do what they contemplated doing with this software. A common license is the GPL, and we've reviewed what the client is doing in comparison with what's allowed under the GPL, and then advised them how to comply with the license, and what's available to them under the license.

Other advice we've given related to negotiating proprietary licenses from open source authors to obtain a license that is different from the standard license.

16. When closed source and open source software are combined, is the resulting combination deemed open source or closed source?

When talking about licensing free and open source software, there's a paradigm shift required. It's not a matter of labeling the code open or closed. We need to go back to copyright principles. If I take someone else's program and modify it such that it's a derivative work, then in order to distribute or make copies of that derivative work, I still need permission.

If clients want to keep their added code separate, such that they can have a proprietary license, than they need to ensure they are not creating a derivative work of the open source software unless they have a license that allows proprietary redistribution. Essentially, we would advise a client not to make modifications to a copyright holder's software unless the client is able to comply with the copyright holder's license requirements. We often advise clients on what activities count as a derivative works.

17. Is open source software in the public domain?

No. Public domain works are works in which nobody can assert copyright rights. With open source software, the authors, employers or assignees retain the copyright and have the right to sue for copyright infringement, just as with software licensed under proprietary licenses.

18. Many companies with large investments in software development are entering the arena of free and open source software. How has this arrival of large corporate interests affected the development of open source software?

It gets taken more seriously because the stakes are larger. When you talk about larger development, more care needs to be put into making sure that the issues are resolved ahead of time.

19. Do you foresee an increase in litigation as open source becomes ever more prevalent?

More copyright litigation can definitely be expected .

20. What are some of the "big cases" that have defined and are defining boundaries in the free software and open source software areas?

The SCO v. IBM lawsuit is a monumental case, so it gets considerable coverage, but that case is only tangentially related to free and open source software licensing issues.

One case where actual terms of the GPL were at issue is the MySQL v. Progress Software case, in which mySQL released software under the GPL. Progress Software allegedly distributed the plaintiff's software without source code and while redistribution is permitted under the GPL, the source code must be provided as well. Ultimately, the parties settled. Unfortunately, there's not a lot of case law in this area because so many of these commercial issues get worked out prior to trial.

21. What are the issues for companies that use open source software?

Companies need to identify what open source software they are using and whether they are in compliance with all the licenses. If a large company buys open source software to run their desktops in-house, and they do not sell that product, there is probably not a danger of violating the license. Often, however, companies are at risk if they incorporate open source software tools into a product they end up selling. Companies will need some mechanisms and procedures in place to keep track of what free and open source software is being used. To keep abreast of compliance issues, in-house counsel should look to the company's program managers and engineers and ask what they included in any given software product.