Derek Skeba
CS 361
OSS Paper

Development Using OSS

There are many ways to make use of open source software. First it is important to look at what exactly open source software really is. It can come in many different forms. It could be in the form of the actual tools and compilers you use to develop your application or it can be in the form of open source libraries your use to integrate and power your application you are developing. Sometimes open source can be as simple as openly distributed source code. This could include simple tutorials or source you find online to show you how to perform a certain aspect in your application.

Open source tools come far and wide, in many different forms as well. There is an open source compiler for almost any language you can imagine. For example, in the C programming language there is the GCC compiler but also Borland, Digital Mars, Eclipse and many more. Most of these compilers have ports for multiple operating systems including Windows, Linux, MacOS, and others. More specifically, even the GCC compiler we may use for compiling our C source code has several ports for almost every operating system. For Windows alone, we can simply go to Source Forge located at www.sf.net and type in "gcc windows." There is GCC for windows and MinGW. These two ports for Windows do not even include the fact that we could try running GCC through Cygwin. As you can see it is not hard to find an open source project for a compiler for any language or any operating system. Even further, you could extend this principle to finding an IDE or Integrated Development Environment for the compiler of your choice. Some compilers come with IDE's attached.

Finding these open source tools is easy in itself. Once you know what you are looking for in your compiler or IDE, it is a matter of going out and locating it. This could be as simple as using any search engine, such as Google, and typing in the compiler you're looking for. For example, "open source c compiler," will give us plenty of relevant hits to open source compilers. An additional website, such as used in the previous example above is Source Forge. This website is an open source community website that allows people to start and list open source projects. By searching for any kind of compiler or IDE on this website, it is guaranteed to be open source software. This website is located at either www.sf.net or www.sourceforge.net.

Moreover choosing your open source compiler or tool in general, is going to be based on several factors. Many of these factors may just be personal preference. However, it is important to use open source software that is peer reviewed, has been around for a while, and is stable. It is usually unwise to go ahead and just grab the first open source tool you find. You want to ensure it has had time for bugs to be worked out and had time for peers to critique it and bring it up to standards. For the most part, when searching in Google, results will be filtered by PageRank so you do not need to worry about this. In Addition, Source Forge gives statistics in their search results for activity and popularity, which may also help guide your decision making (SourceForge). For the

most part, it is just important to do a little research and ensure the open source tool you are looking to use is well developed and is not going lose support too quickly.

Moving beyond just open source tools, let's take a look at open source libraries and components. Open source libraries are what we may use to power our application we are developing. Many times regardless of the application type, we do not have the time for developing every technology in it. Or there is simply a library that is already up to standards, which can fulfill our needs. Rather than going through the work to develop and test your own library you can save a decent amount of time by researching and using an open source library to get the same work done. These open source libraries, again, like most open source software, has had time to be peer reviewed and developed up to industry standards. So in this fact, you know you are already getting a quality technology. Many open source libraries may already be in products you know today, and can therefore be trusted a little more.

Let's use the example of a 3D modeling program. Spending the time to develop a 3D graphics library in OpenGL or DirectX can be quite time consuming, especially if we want to port it to both OpenGL and DirectX for full user compatibility. In this case, there is an open source library already available that can do just that. OGRE is a 3D graphics library that is object oriented and works regardless of implementation; either OpenGL or DirectX. OGRE is just one example, of an open source library we could use for our 3D modeling application. This brings us to the topic of where we can find open source libraries.

Finding an open source library is almost the same as trying to locate open source tools. A search engine like Google, again, can be used to easily locate popular libraries. Source Forge is also a great source for finding community driven development. An additional website I find useful, is www.thefreecountry.com. Here they have lists of many libraries, as well as open source tools, that you can use for free in any of your projects you are developing.

Integrating an open source library or component could possibly be the most important step in using open source software in your application. Many times open source libraries will give us a major advantage over self development, in that they tend to be very powerful for little effort put into them. In other words, it only takes a little time to integrate a libraries API (Application Programming Interface) and they usually get a lot of work done we would normally have to do ourselves. The first step you should take with any open source library is to become familiar with the documentation and maybe take a few basic tutorials to gain the knowledge needed to implement it. After that you're more in likely going to want to try installing a SDK (Software Development Kit) or something similar, if one is available. Once you feel like you have a grasp on how everything works, you can probably begin writing your code that will use this library.

One possible method of integrating your open source library would be to separate any library or API specific code in its own module or separate source file depending on the language you are coding in. For example, if we were programming in C, you could

write a library specific file called “ogre.c” and then include this in all of the other files or main header file. By this method, all your other application code would have access to your technical methods that use the libraries API and would maintain some structural organization.

Aside from structuring your library specific code in its own module or source file. A majority of the time, much of your applications code can be written without worrying about the open source technology you plan on implementing. Using the 3D modeling application as our example, we could easily design the main code leaving gaps to use our library later. We could even start on things such as user input, our GUI, and error handling before we even write a single piece of our 3D animation or viewing code.

Let’s tie this all together using our 3D modeling example. The OGRE website (www.ogre3d.org), provides its very own SDK which is where we would probably want to start. On the SDK page, we can see that this library only supports C++. This is a piece of information we would be interested in knowing before deciding to use such a library. Once we download the SDK we should familiarize ourselves with the documentation it provides and maybe even some simple tutorials. Continuing on, the website even provides an API reference which would be very useful when writing and implementing our library specific code in our application (OGRE). Although this is just one example, most widely used open source libraries tend to provide at least these basic features, which can make implementing your own library much easier in the long run.

After you have completed your software which uses an open source library, you will probably ask yourself two things: can I sell this software for a profit and can I copyright this software? Most users find this topic confusing and ambiguous. In fact, it is usually very easy to understand. The core of open source software lies in its licensing. While there are in reality hundreds of different open source licenses, there is one license that is used in far majority over the others. This license is the GPL or GNU General Public License. Whenever software is licensed under the GPL, there are three major rights that are granted to all users who receive the source code. Firstly, “anyone can use the code anywhere in any situation”, secondly, “anyone can redistribute the code to anyone else, as long as the source code is included and the distribution license remains the GPL”, and thirdly “anyone can create a derivative work of the code and redistribute it, as long as the resulting source code is also made available at redistribution time, and as long as the resulting source code is licensed under the terms of the GPL” (Adida).

By now you may be asking yourself, what does this mean in terms of making a profit and copyrighting my software? Easily put, you can sell and choose to make a profit off your software. There is nothing stopping you from selling your software for a small fee or even a very large fee. As long as you provide the original or derivative source code for the open source library you used with it. Even if you choose to modify the source code to the library, you can sell it, but you must distribute the source openly to anyone that receives your software. Generally, open source software is not sold for profit, because since the source code must be distributed along with it, other users could choose to give that source code away for free, under the GPL or similar licenses. It is important

to note, you do not need to distribute the source code to any piece of the software you may have written outside of the open source library. For example, in our 3D modeling program, you need only to distribute the open source library, there is no need to distribute the source code you developed for taking user input, or displaying your GUI.

Moving on to the second part of the question; can I copyright my software? There is nothing in the license from preventing from doing this. You may copyright code you write that is external to the library you are using. In essence, you can copyright your 3D modeling application which uses the open source library; however, even if you make a derivative version of the library, you may not copyright the library itself. The open source library is protected under the GPL license, and all derivative works as well. As we can see in our example using the OGRE open source library, it is licensed under the LGPL. The LGPL is a slightly modified version of the GPL and for the most part works the same. You can sell your software that uses the OGRE library, and you could choose to copyright it just as well. However, in any case, you would need to provide the same LGPL license to the user and also provide the full source code the library you used.

In my opinion, the legality of these open source licenses is quite sound. Developers should be able to make their own decisions as to sell for profit and/or copyright their software which uses open source components. A majority of open source licenses make conditions unfavorable for choosing to do so. This in a sense keeps the original work of software or open source component free software. In my opinion, free software should not mean software that costs no money, but software that promotes freedom of knowledge and freedom of source code. If the source code is provided another developer or group could just as well make a similar piece of software, which will therefore decrease demand for the original software. This helps keep a ceiling on how much developers can charge for software using open source components and I think this is all the protection that is needed and is inherent in licenses such as the GPL and LGPL.

Works Cited

Adida, Ben. “Understanding Open-Source Licensing.” Open ACS. Retrieved on March 27th from http://openacs.org/about/licensing/open-source-licensing.

OGRE. “OGRE – Open Source 3D Graphics Engine.” Torus Knot Software Ltd. Retrieved on March 27th from http://www.ogre3d.org/.

SourceForge. “SourceForge.net: Open Source Software.” SourceForge, Inc. Retreieved on March 27th from http://sourceforge.net/.

The Free Country. “thefreecountry.com: Free Programmers' Resources, Free Webmasters' Resources, Free Security.” Christopher Heng. Retrieved on March 27th from http://www.thefreecountry.com/.