As we all know, software engineers usually implement and distribute software in discreet modules or files. Typically, each of those modules is copyrighted by its authors and licensed to others. Open source projects, like Debian, collect such modules and distribute them as aggregations, sometimes called distributions or packages. Downstream companies and individuals make modifications and distribute the resulting modifications in a similar manner. By the time software arrives on your computer (for example, on a disk or as a download over the Internet), it is often an aggregation (or collection) of many modules from many places. When installed on a computer to create executable applications, the applications interact with data and business processes to perform an enormous variety of activities.

Then there are software patents. Consider the world of software patents as a giant and ever-expanding bag of patent claims, for example, method claims covering methods that can be carried out using software and/or apparatus claims that cover apparatus including software as a component. Inventors around the world are constantly dropping new claims into that bag.

One question you may ask is: How do patent claims in the bag relate to an aggregation of modules?

It is often the case that an author or copyright owner of a module isn't the inventor or owner of a patent claim that covers a method that can be carried out using that module. When the copyright and patent owners are the same, you can just read any patent terms in the copyright owner's software license for rights on at least some (but probably not all!) of that licensor's patent claims. We assume here, however, that the copyright and patent owners are different people or companies, or that some of the copyright owners' patent claims aren't licensed with the software, because that makes the analysis more interesting.

A patent owner can prevent you from making, using, offering for sale or selling a patented invention within the United States, or importing the invention into the United States, 35 U.S.C. 271(a). This is true whether the software is original or copied. This is direct infringement.

It makes no difference to an infringement analysis whether infringing software is formed from a single module or is formed by linking or combining software from two or more modules in an aggregation of modules that comprises a software system.

Patent law also covers situations where one person makes and distributes only some components of a patented invention (i.e., where such components correspond to elements of a claim). For example, one can be liable as a contributory infringer if one offers to sell or sells within the United States or imports into the United States a material or apparatus for use in practicing a patented process, "constituting a material part of the invention, knowing
the same to be especially made or especially adapted for use in an infringement of such patent, and not a staple article or commodity of commerce suitable for substantial noninfringing use.” 35 U.S.C. 271(c). Indeed, in certain circumstances one can also be liable as an infringer by supplying, in or from the United States, components of a patented invention so as to actively induce the combination of such components outside the United States. 35 U.S.C. 271(f).

When a computer user uses software to carry out a method covered by a claim, the user of that software infringes the claim, and distributors of the software may also be liable as infringers.

For most method claims of software patents, an infringing act (i.e., direct infringement) takes place on a user's computer which is under the control of that user while processing the user's instructions and data. However, the patent statutes we just cited make it clear that others involved in the software distribution business who supply the components of infringing software can also be held liable as infringers.

Imagine that one of the software modules in a large Debian distribution infringes a claim in the bag of patent claims that Debian doesn't own.

Infringement can be avoided by removing the infringing module, and that may be necessary even if removing a single infringing module makes the entire program stop working. Infringement can also be avoided by substituting a different module that doesn't infringe because it provides different functionality. But note that merely substituting one version of software for another may not avoid infringement if both modules provide the same functionality covered by the patent claim.

Patent claim infringement analysis regarding a particular software module does not impact other modules in an aggregation unless, perchance, some combination with one or more of the other modules (for example, when linked together) infringes one of the claims in that bag of claims. It may even be that the original module alone doesn't infringe at all, but only when that module is combined with other modules in the aggregation (e.g., linked statically, linked dynamically, linked any way you can imagine!) is there an infringement of a patented method claim. In that situation, the cure for infringement may be removal of one or more of the modules that collectively result in an infringement, or changing the program so that it doesn't perform the claimed method.

Note that claim infringement analysis has nothing to do with any copyright relationship among the modules in an aggregation. Indeed, claim infringement may not involve modules at all; it may be that only a subset of instructions within a module performs a claimed method and needs to be dealt with, or that part of the claimed method is performed in hardware and only part in software. The infringement may even involve a combination of modules with specific data, or in a specific applications environment, or in response to specific user commands.

This is the reality of patent infringement in the world of software. Copyrighted software can infringe software patents in ways that may require surgery of functioning software—or prevent its use altogether.
Given the huge numbers of claims in the worldwide patent bag, matching patent claims to modules and aggregations of modules is a time-consuming and expensive task.

Companies that distribute hardware are very familiar with the patent system. The added burden to them (in technical human resources and dollars) of analyzing combinations of patented software and hardware is typically built into their business models already. However, when companies distribute software that has many possible uses when installed on downstream computers, preventing or predicting patent infringement liability is more difficult and expensive since liability may depend on how the software will actually be used.

This is a burden that we all want to reduce or eliminate from the free and open source software development process. That is why we seek licenses or covenants to patent claims, so that at least some of the claims can be removed from the bag for our practical purposes.

For traditional companies with large patent portfolios, it makes sense to avoid the problem of matching patent claims to software modules by cross-licensing their patents with other large companies. Huge swaths of patent claims can be eliminated from consideration—if you are large enough to own a patent portfolio worth trading in the first place. This solution isn't often available to smaller companies. Furthermore, this solution does little to promote free and open source software except in rare situations when it is in a large company's financial interest to shield a specific open source project.

The International Characters Covenant Not To Assert addresses the complex patent-matching problem in a different way. If you do not want to concern yourself with matching your software modules to any patents subject to a covenant like the IC Covenant, then make sure your software is actually distributed to the public (by you or someone else) under a free or open source license. For those readers not familiar with the International Characters Covenant, see www.rosenlaw.com/IC-Business-Model.pdf.

Under the IC Covenant, companies that, for their own commercial business reasons, want to be selective in the disclosure of their software will need to undertake the extra effort to make sure that only those modules or combinations of modules that infringe IC’s patent claims are disclosed. And if companies distribute or use proprietary software that performs IC's patented methods, or distribute hardware combined with patented software, a commercial patent license is required. The IC Covenant places the burdens and costs of patent infringement analysis on those who obtain commercial value (and presumably profit) from patented software-based products and services, while eliminating patent infringement risk for developers and distributors of free and open source software.