Software Patents: Why Should We Care?

If nature has made any one thing less susceptible than all others of exclusive property, it is the action of the thinking power called an idea, which an individual may exclusively possess as long as he keeps it to himself; but the moment it is divulged, it forces itself into the possession of every one, and the receiver cannot dispossess himself of it.

—Thomas Jefferson

Imagine a world in which Socrates not only originated the dialogue form of inquiry but also filed a patent claiming “intellectual property” rights for his invention and then vigorously enforced his patent against the Sophists to ensure the “purity” of his learning method. Imagine an Oxford scholar traveling to the University of Bologna in the thirteenth century, observing the “lecture” style of teaching, adapting it for use at Oxford, and then filing for patent protection. A cross-licensing agreement permits Bologna faculty to use the lecture format, but Cambridge University faculty are locked out. Imagine an MIT faculty member today taking out a patent for a novel application of “game theory and team-based problem-solving techniques” in physics laboratory instruction. A dispute soon erupts when Harvard claims that the underlying idea of the patent was derived by the faculty member when she was previously employed at Harvard. Finally, imagine a world in which individual faculty, educational institutions, and for-profit companies rush to stake out patent claims for every conceivable teaching style and method, competing to develop the largest and most aggressive patent portfolios, based in part on the ingenuity of the faculty and in part on a series of strategic acquisitions and cross-licensing agreements.

These imaginary examples from a parallel universe likely strike us as fanciful if not absurd, for in our world, teaching methods are simply not patentable. Or are they? As the world becomes increasingly digital, as more educational technology tools become commonplace, as more faculty utilize cyberspace for some part of their teaching, and as the boundary between teaching in the traditional classroom and teaching online begins to blur, our intuitions about property, collaboration, and academic freedom will be put to the test. In our universe—and not just in the parallel one I have described—there is a rush under way to carve out and “own” the basic building blocks of ideas that enable all educational technology. The prospector’s gold this time around is the software patent.

Software and Intellectual Property

Let’s quickly review the software development process in order to understand where intellectual property enters into the picture and how the case law has evolved radically during the past decade. The basic process of developing software is similar to that of building a house. Step one in building a house is the idea phase: specifying the house’s design. The design, in turn, is used to construct a building, which represents the physical embodiment or expression of the architectural idea. Finally, a set of laws governs the appropriate use of the building.

All software, as a first step, originates with an idea or conception. As a second step, the idea is given physical expression in the form of computer code. And as the third and final step, the author of the code attaches a license governing the code’s distribution, use, and modification. These three basic steps underlie all software development.

Until very recently, the assertion of ownership entered into the picture only during the second and third stages of the software’s lifecycle. Copyright, which is one form of intellectual property protection, applies only after a programmer has given the design ideas physical expression in the form of code. You and I could take the same design ideas as a starting point and program them independently. I would then own my code, and you would own your code, but neither of us would own the ideas on which our respective software was based. The code—the physical expression of the ideas—was and still is protected by copyright. If I were to steal your code, I would be in violation of the copyright. The ideas and methods, however, were until recently freely available to all.

Software patents, which protect ideas and methods, are a relatively new development. The U.S. Patent and Trademark Office (USPTO) had previously been reluctant to grant patents for software. The case law began to change radically with a set of Supreme Court rulings in the 1980s and a federal circuit court case in 1998 (State Street Bank & Trust Co. v. Signature Financial Group). These court rulings opened up the software patent floodgate. More specifically, the State Street Bank case upheld the notion that “business methods”—that is, ideas—are patentable subject matter.

This shift in the case law means that the ideas and methods underlying software can now be protected and assigned ownership. The Blackboard Inc. v. Desire2Learn Inc.
case is significant in this regard because the dispute is not over copyright infringement. Desire2Learn has not been accused of purloining Blackboard’s computer code, which deserves protection and is protected by copyright laws. Blackboard was issued a patent for an idea that can be expressed quite simply: “We conceive of a course-based system in which a user may (a) have multiple roles (i.e., the same user could have a faculty role in one course and a user role in a different course) and (b) be presented with course information drawn from different courses in a single home page.” Blackboard’s patent claim is breathtakingly broad and sweeping, for it potentially covers a core design idea underlying a wide range of learning environments. Setting aside the question of the legal validity of Blackboard’s patent, we should be aware that Blackboard has been granted a twenty-year monopoly over all software that implements the idea expressed in its patent claim.

The Problem with Software Patents
Oliver Wendell Holmes noted that the law can be viewed as a “reaction between tradition on one side and the changing desires and needs of a community on the other.” Academic freedom, a bedrock value of our community, means in part that ideas can be shared freely. For example, no individual or organization “owns” the Socratic method or the case method for teaching law or management. Faculty routinely “rip, mix, and burn” teaching styles with their colleagues. When it comes to teaching methods, academic practice and the law are mostly in sync with one another.

In the realm of software, however, we are beginning to see a widening gap between the beliefs and practices of the academic community and the law. In order to better understand this gap, let us return to the parallel universe and see how this gap would look if it concerned teaching methods rather than software. Let’s say that in the parallel universe, the laws permit ownership of teaching methods as a patentable business method. What criteria could we use to ensure originality and novelty? Where does one idea begin and another leave off? How would competing claims be adjudicated? How would the Athenian Patent Office determine whether Socrates developed the Socratic method himself or whether he picked it up and adapted it from others? How would variant ideas be distinguished? Or, in a modern-day hypothetical example, is the case method as used by a faculty member at Harvard Business School the same as the case method as practiced at Yale Law School? Is it even the same method when used by two different faculty members at the same institution? These questions would be intractable in the parallel universe, and they are just as intractable when applied to software in our real world.

The statements and actions of the administrators at the USPTO reveal that the administrators recognize the problem but also know they are powerless to address it. Their response has been to throw the entire problem over the wall to the courts. This means that as the number of software patents grows by thousands each year, the validity of individual patents will be decided not on their merits but as a result of time-consuming and expensive litigation.

Toward a Solution?
I am not sanguine that Congress, the USPTO, or the courts will address the software patent issue anytime soon. Is there something we can do, in partnership with the business community, that meets the community’s needs but that is also in line with academic beliefs and practices? I think there is. I propose that we consider a collaborative model of developing educational software—a model based on disclosure and peer review. This is not a new model but rather is an application of the open-source methodology at the idea stage of software development. The higher the level achieved, the more “compatible” the company is with academic values.

Level 1: Basic Disclosure. The company publishes a list and a description of its educationally relevant software patents, both those it currently holds and those it has applied for.

Level 2: Peer Review. As proposed recently by IBM, the company puts its patent filing online for public comment, review, and scrutiny for prior art. The company also allows for a legally nonbinding peer review of its education software patent claim by domain experts. The review process could be mediated by an entity such as EDUCAUSE.

Level 3: Pledge. The company declares that its intention in holding the patent is strictly defensive. It will not file for patent infringement or expect royalties from any party unless it has been threatened by that party.

Higher education is at a crossroads. As educational institutions face rising costs, declining public support, and increasing competition from for-profit entities, information technology is one of the few strategic levers we possess for controlling costs and improving quality. The specter of software patents has now placed that strategic lever at risk.

Alfred H. Essa is Associate Vice Chancellor and Deputy CIO for the Minnesota State Colleges and Universities.