

Award-winning paper delivers stark critique
Student argues software is patentable

(Law Times, September 22, 2003)

By Daryl-Lynn Carlson
For Law Times

A third-year law student at the University of Toronto has offered up a comprehensive critique of problems protecting computer software.

Christopher Heer suggests in a research paper that copyright protection should be limited to literal infringements, and non-literal elements of computer software would be better protected under patents.

Heer's argument and research has garnered him the first-ever Stikeman Elliott LLP prize for best technology paper offered through the university's Centre for Innovation Law and Policy. Heer receives \$2,000 and a mentorship to rework the paper for publication in a legal journal.

The paper provides some clarity to protecting software other than what has been established under current provisions contained in copyright legislation, which has proven inadequate and has been subject to numerous challenges.

"What the courts have been struggling with is how do you protect something as a literary work when it really is very different from a literary work," says Richard Brait, head of Stikeman's technology group.

Brait notes software is "very much driven by functionality and what it can do in a technological sense and traditionally copyright hasn't applied to that. It isn't supposed to generally apply to ideas or things that produce functional results.

"This is one of the key areas of intellectual property law generally that is in need of some clarification," Brait says.

In his paper, Heer goes into great technical detail to explain the design and functions of computer software.

He acknowledges in the paper's introduction: "Some elements of computer programs, such as the menu command hierarchies in the user interface, are methods of operation that properly fall within the domain of patent law.

“Other elements, such as screen displays of the user interface, are graphic works and thus subject to copyright law.”

But he says, “While these classifications seem logical, the system of protection that has been extended to non-literal elements of computer programs is problematic.”

The non-literal elements commonly refer to elements of a computer program’s structure and design.

In the early 1990’s, Canada adopted a test to determine non-literal infringement based on the United States Court of Appeal’s 2nd Circuit Court decision in *Computer Associates International v. Altai Inc.*

Called the abstraction-filtration-comparison (AFC) test, it is used to determine copyrightable and derivative works of software. Heer argues the test “has confused the courts and is very much misapplied.”

Moreover, he suggests, it is in the application of the merger doctrine, which eliminates elements of functionality from copyright, to computer programs where “the AFC test falls apart.”

“Artistic literary works can be purely creative, yet computer software is essentially utilitarian. Since computer programs are directed towards practical application, their creation involves technical skill as opposed to artistic expression,” he says.

“When this concept is applied to computer software, those elements of computer software that are necessarily incidental to its functions are not protectable.

“This presents a huge obstacle to the protection of computer programs.”

Further on, Hess says the creation of computer software is driven by efficiency concerns and affords few options for design expression by software designers simply because they are restricted by their mandate to make the software work.

Throughout the paper, Hess refers to a number of court challenges and courts that have acknowledged confusion over how to treat software in an intellectual property context.

“Even Judge Walker recognized in *Altai* that ‘to be frank, the exact contours of copyright protection for non-literal program structure are not completely clear,’” Hess mentions.

At first glance, writes Hess, computer programs appear to be textual works, with their source codes comprised of limited and perhaps awkward vocabularies.

“This aspect suggests that we can extend well-established copyright principles from the literary domain to that computer software. Yet the essential aspect of computer software is what it does and how it interacts.”

In that context, Hess concludes that software is better protected through patents than copyright.

“Given the apparent functional/non-functional dichotomy between patent and copyright, and accepting that computer programs are functional, it follows that the design elements of a computer program should constitute a patentable subject matter just as the design elements of a mechanical invention can be claimed as such,” he writes.

“*Altai* has been viewed as a landmark decision as it incorporates many traditional principles of copyright law into a single analytical framework seemingly suitable for computer software, he concludes.

“However, when honestly applied, the AFC test eliminates protection for computer programs. . .”

It appears, says Hess, that the U.S. legislature ultimately made a “critical error” when it developed copyright provisions for software by viewing computer source code analogous to a literary work.

Rather, closer scrutiny would result in software being likened more to “the gears, pulleys, and levers of a mechanical invention” because of the limits on creativity in order to be functional.

“When viewed as a collection of software objects combined in such a way as to optimally perform various tasks, the design of a computer software closely resembles the design of functional devices protected by patent law rather than the non-functional, non-literal elements of creative authorial works protected under copyright law,” Hess concludes.

His 25-page paper can be read in its entirety under the Working Papers section of the centre’s Web site at www.innovationlaw.org.