1. There is a current discussion on immigration reform. Several high tech firms are pushing for opening up our borders to everyone with a high tech degree (phd or lower). There are arguments about the benefits to having this influx of technical people. Others argue loss of jobs for Americans. Do you think immigration reform is needed? How would current proposals affect the high tech industry? What are the risks and benefits?

It is our belief that immigration reform is needed. We take this stance for the following reasons:

- A company must first prove that a job offered to an immigrant was not lost to an American.
- A company must apply for a work visa with the following conditions.
    - There are approximately 80,000 work visas issued every year.
    - With an additional 20,000 work visas designated to post graduate students.
    - There is also an approximate cost of $10,000 on the company's part, plus the cost of an experienced employee in immigration law.
- After a company has gone through the process of issuing a work visa for an employee, the employee is under no obligation to stay with the company.

For these reasons, we believe the system that is currently in place is designed to prevent employers from considering immigrants for positions in their company. We see this as a problem because we currently invite approximately 300,000 graduate level international students to learn and be trained in the United States. After they have received a degree, if they are not able to gain employment, they are deported back to their country of nationality. These is a problem because these are dedicated individuals and trained in the United States, then are not allowed to contribute towards the American economy.

We believe that by releasing more work visas and reducing the investment on the company's part needed to obtain a work visa would allow more highly qualified employees to enter the job market. By keeping the requirement that a company must first prove that a job offered to an immigrant was not lost to an American, insures that no job is lost to an American citizen. Also, by allowing for more highly qualified employees to enter the job market it would allow for more advancement in the high tech industry. For these reasons we believe that immigration reform is needed.

2. What do we mean by open source? What are the different forms of open source and how is it defined/regulated? What role will open source initiatives play in the industry in the next decade? What is their impact on the industry? There are very vocal proponents for open source; in that it is morally the only course of action. Others argue that it is not a good economic model. Is open source the correct answer?

Open source software is software with source code that is freely available and redistributable. There are two general categories for open source licenses: copyleft and permissive. Copyleft licenses require that software is uses the same license as the software it is based on. This means that all software based on a project with a copyleft license, regardless of how distantly, is published under the same license as the original project. An example of a copyleft license is the GPL. Permissive licenses are licenses that allow derived works to use a different license than the work they were derived from. This allows a proprietary program to use open source libraries. An example of a permissive license is the MIT license.

Open source is good for software libraries. Software libraries are bundles of useful and reusable code. They tend to be useful in creating a wide variety of products, but do not do anything that end users would find very useful. If a company develops a general purpose library for its own use they limit the amount of people who can work on it and waste their employees' time by having them write something that could already exist as an open source library. Open sourcing libraries allows more programmers to write, test, and maintain them, which results in a more stable and highly developed library. They give people writing a program for a specific purpose existing code that they can use to take care of a lot of commonly done operations, like talking to a server or parsing XML. Using libraries for these tasks instead of writing the code from scratch, can save companies a lot of money and allow independent developers to actually develop their software rather than getting bogged down with details.

However, open source is not a reliable way of making money. If a company sells open source software, somebody else can sell the same thing for less or give it away for free. Developing software is hard and time consuming, and the people or companies that develop software have the right to protect their work by keeping their source code private. Some people say that if people can't look at the source code of a program they use, they can't really trust it because they can't verify that it's only doing what the company that developed it says it does. We argue that if someone can't trust the company that is making, that person shouldn't be using that companies software. Expecting to see the source code for all of their programs is like somebody expecting to examine an airplane before the fly somewhere. People trust that the company the built the plane and the airline the maintains and operates it know what they're doing and aren't trying to kill their passengers in a fiery crash.

3. Are current IP and patent laws sufficient for the software industry? Why/Not?

Current IP laws are actually inhibiting the growth of software in the general sense, and network applications in the more general sense. There are a few reasons for this:

1. Patent and copyright terms are fundamentally too long for Computer Science-related topics. Today, standards are born, mature, and die sometimes in as few as five years. Today, with few exceptions, almost all software and hardware from 20 years ago is not only out of standard use, but is all but impossible to use. Printers from the early 1990s and late 1980s used LPT or serial connections, for example, which are no longer present on most personal computing devices. Dial-up modems have been replaced with 802.11-compatible wireless cards. Giving a patent a term of 20 years after filing and copyrights of 90 years after the death of the author prohibit a large variety of devices from being used as the basis for other inventions.

2. Computer Science builds on itself inherently. This is beyond simply having a mechanical system which can be driven off steam instead of cranked by hand, or improving the material in a gear: When a new algorithm is figured out, it can usually be used in a very wide variety of applications which bring previously unreachable goals into focus. The best example of patents hampering computer science advances is that of the Java language. Strictly speaking, the language is the property of Oracle (Previously Sun Microsystems), and cannot be used outside of their proprietary systems. Although there are implementations of Java outside of Oracle's JVM, these leave the creators of the modified VMs succeptible to litigation, as seen recently with Oracle vs. Google in the android context. This litigation has a chilling effect on developers who might otherwise be free to innovate, and reduces productivity of software engineers

3. Patents, at least, are not really applicable to software. A patent applies to devices or hardware, physical objects that are unlikely to be created in the same general form without some outside influence. Software, by contrast, is created through a series of expressions, which could be just as easily generated by pointing a chimp at a punch card typewriter and letting it go to town for a sufficient period of time. It is unlikely, to say the least, that a similar uninformed humanoid could reproduce spontaneously the schematics for a cotton gin. This makes it entirely too likely that parallel concurrent development will result in multiple legitimate cases of nearly exactly the same algorithm or set of instructions, making a patent inappropriate.

Copyright is much closer to being applicable, since a program is generated through a human-readable set of code in almost all cases, and the human-readable code has almost exactly the same definition as poetry: Meaning, given a fixed form in language for a specific purpose. The fact that the languages are based upon mathematic logic rather than some abstract need to communicate about surroundings does not change the nature of the source code, and thus it

SHOULD fall under copyright. As mentioned above, however, software does not usually survive more than a couple years before being completely amortized (depreciated). 5 years is considered a long time in Computer Science. Consider Windows XP: Although still in use, there is no evidence that Microsoft even considers the program worth supporting or selling. Windows Vista, despite being a black mark on the company's name, is almost in the same boat. Both have fallen to such a state after less than a tenth of the lifetime of copyright.

Trade Secrets are probably the most appropriate use of IP for Software. Because the exact source is going to contain a great deal of human optimization, it should be the responsibility of a company to keep its source from wandering home with the wrong person after the company New Year's party. People reverse-compiling source will still have quite a bit of work to do to get the source into a usable state, and producing exact replicas with reverse-compiled source is copyright infringement (as above), so there is no reason that such should NOT be treated as a trade secret.

Trademarks could also apply to the design of an interface (The look of a top bar in Windows XP, with the three buttons on the top-right side, is a good example), but should not apply to code inside any more than a trademark would apply to a poem or general instruction packet. Whether or not a trademark can reasonably apply depends on exactly what software is being considered.

In direct answer to the question, Yes, some elements of IP law are relevant to Software, but almost all of them need some serious legislative changes in order to be helpful to the community in general.

Should changes be made to patent law?
Yes, Software should be explicitly excluded from patent law, specifically. Patents do not mesh well with the actual nature of software. Just because it can sort a haystack and show you the needle doesn't mean it's the new version of the cotton gin. To properly address the uniqueness of software we need a new class of patents (see below).

Should we have a new arena of software patents with different IP law?
The ideal solution would be to simply eliminate patents. Due to the short life cycles of software copyright gives creators enough of an advantage. However, in practice this is not possible. Before software patents IBM would patent hardware that would closely resemble the software they were trying to patent. We do not want to leave this avenue open so that companies and attorneys can exploit existing law. To hold a software patent you must show a specific solution to a problem. This will allow other solutions to coexist with your patent. A software patent will only last 5 years from issue date. Finally a software patent may not be traded or sold.

How can one affect changes to IP law?

Assuming one is affecting changes in the United States, they key is to petition Congress. The current Patent Office was established under the State Department by Congress during a review in 1836, and is regulated by Congress.

Can this be enforced globally at all, and is there a negative consequence to the United States if such measures are taken?
We have a hard enough time getting international agreements on rape and cartoons. I doubt sincerely that if things of pretty clear importance can't be enforced globally, that there will ever really be an agreement on something so trivial as the entitlements given to those who write software, which is by its nature easy to copy and distribute. About the only way for EVERYONE to do that is to take some seriously draconian measures with copy protection, which would probably destroy a lot of value from software and entail some extraordinary efforts on the parts of the best minds in Computer Science, or to have one organization dominate the globe and impose the same strictures on all of the world. Neither speaks well of relations between the U.S. and any other entity.