THE BROOKINGS INSTITUTION

SOFTWARE AND LAW:

IS REGULATION FOSTERING OR INHIBITING INNOVATION?

Wednesday, December 7, 2005

3:00 - 5:00 p.m.

The Brookings Institution
Falk Auditorium
1775 Massachusetts Avenue
Washington, D.C.

[TRANSCRIPT PRODUCED FROM A TAPE RECORDING]

MILLER REPORTING CO., INC.
735 8th STREET, S.E.
WASHINGTON, D.C. 20003-2802
(202) 546-6666

Panelists:

BRIAN KAHIN
Senior Fellow, Computer & Communications Industry Association; Adjunct Professor, University of Michigan School of Information

DANIEL B. RAVICHER
President, Executive Director
Public Patent Foundation

EMERY SIMON
Counselor, Policy Council of the Business Software Alliance

KENNETH DAM
Senior Fellow, Brookings Institution
Max Pam Professor Emeritus of American and Foreign Law, University of Chicago Law School

Moderator:

BEN KLEMENS
Guest Scholar, Brookings Center on Social and Economic Dynamics

P R O C E E D I N G S

MR. KLEMENS: [In progress] for a structured document browser. So to give an example to this sort of thing, let's say you go to house.gov, the House of Representatives website. On the left you'll see a list of links. You've all seen websites like this that say, you know, "Click here to go to the representatives list," "Click here to contact your representative," "Click here to see the committee list." And then the main page has various changing, you know, a document depending on what you click on.

So Ameritech was granted a patent on a structured document browser that fits this description, that has a fixed frame and unmutable main window that delivers information. Ameritech turned into SPC, which I guess is now AT&T. So house.gov is violating AT&T's patents on the face of it. This is not entirely academic. SPC demanded royalties from a number of companies, such as an educational tour company named Museum Tours.

Or let's take this a step further. When you visit house.gov, you load this page onto your computer. Let me read to you from In re: Alappat, a 1994 ruling, which held that such programming creates a new machine because a general purpose computer in effect becomes a special purpose computer once it is programmed. So that means that you are on par with the guys at the house.gov basement. So when you load this software, you as well are infringing this patent.

So the question here is what went wrong? How do we have a system where visiting a website or loading software is a violation of patents? Is it that this patent is over broad and shouldn't have been granted? Is it that the entire concept of software patenting is fraud? Is it that the guys at the House IT Department should have

been doing due diligence, or is it just that we have an overzealous moderator, and you know, this is perfectly fine and this is the price we pay for innovation elsewhere?

With me to discuss these questions we have Brian Kahin. He's a Senior Fellow of the Computer and Communications Industry Association, and an Adjunct Professor at the University of Michigan School of Information. And he recently co-organized a panel on Advancing Knowledge Economy in collaboration with the OECD, European Commission and National Science Foundation, and has actually been organizing conferences on software patents. When was your first—?

MR. KAHIN: 1988.

MR. KLEMENS: 1988, so he's been working on this for a while.

Next in the speaking order will be Emery Simon. He's a Counselor for the Policy Council on the Business Software Alliance. Before he was at the Office of the U.S. Trade Representative, where he was the principal U.S. negotiator on intellectual property for NAFTA and other treaties, and he's also worked for the Congressional Budget Office, the Inter-American Development Bank, and the Council on Environmental Quality.

Ken Dam is a Senior Fellow at Brookings, and Professor Emeritus and Senior Lecturer at the University of Chicago Law School. From 1985 to 1992 he was the Corporate Vice President of IBM, where he oversaw, among other things, patents and intellectual property, and in his spare time he was Deputy Secretary of State and Deputy Secretary of the Treasury.

Dan Ravicher at the end is the Founder and Executive Director of the Public Patent Foundation. Before he had been a patent attorney with extensive

experience in the day-to-day patent affairs of litigating, licensing, prosecuting patents, and he clerked for Judge Rader of the Court of Appeals for the Federal Circuit, sort of a star in the patent world.

And me, I'm Ben Klemens; I'll be your moderator. I'm a guest scholar here at Brookings, and mostly I write code to analyze social network formation and model integration. But I'm also the author of this attractive hardcover book: Math You Can't Use: Patents, Copyright and Software. The press paid for the coffee you're drinking now, so I should give it a little plug. It's a red and silver book, the Christmas winter theme, a perfect stocking stuffer.

[Laughter.]

MR. KLEMENS: As well as being shiny, the book covers a lot of ground in these matters, gives an intro to patent and copyright, discusses some mathematical results about the relation between math and software, and discusses some of the legal history about this question: well, if math should not be patentable, if machines should be patentable, how do we draw a line between them for software? And it gives a ground level discussion of how businesses and the modern day are handling the question of software patents, and an overview of copyright and how it should be applied to code.

So enough about me. Now each of my panelists will have 15 minutes to speak. Della Feher, at the Center on Social and Economic Dynamics has the time cards and a bell if need be. Then I'll open discussion among the panelists, and in the last half hour we'll take questions from the audience.

So without further ado, Brian.

MR. KAHIN: Okay. I'm going to give an overview of some of the characteristics of the patent system today, and how they impact software in particular, and perhaps information technology more generally.

First of all, patents should be understood as a hybrid form of regulation. They're not regulation in the classic sense of having one government agency oversee the functioning of a law in a particular area. They do involve property rights that are granted by the government, and these rights are granted on an ex parte basis, that is, between the applicant and the examiner, a very unusual kind of procedure. This in fact creates a database of private regulations that you can go to and see who got a patent, but that database does not tell you who has it now, under what terms they are available, or even whether in some respects whether the patent is still valid or not, or ever was valid.

These rights are privately enforced. The government does not have a role in enforcing them directly, but they're enforced through private litigation, and this is very costly, as I'll show you in a moment. And this process, this set of institutions is subject to capture at multiple levels, as regulatory systems typically are.

It's resolutely one-size-fits-all, and we have provisions in TRIPS, which is part of the WTO charter, which lays this out, somewhat ambiguously as to what extent you can tailor the patent system, but it basically says patents shall be available in all fields of technology.

And finally, independent creation is not permitted. This is a fundamental distinguishing characteristic from copyright, which allows for independent creation. So this is what the database of private regulations looks like. It's an upward sloping curve as these patents accumulate, so we're really just at the beginning of a phenomenon of a

growing body of software patents. Privately enforced through costly regulation, these are the 2003 AIPLA figures, and as you can see, that when small amounts are at stake, the cost of litigating patents exceeds the amount in controversy. Furthermore, for a small company, you also have to consider that the cost of patent litigation involves staff time, opportunity costs and distraction from business focus. So patents do not scale down well in terms of the transaction cost of the system.

These very high costs results in only 100 patent cases going to full trial each year, and this is a rule of 25 as explained by Chip Lutten (ph) from Apple Computer in testimony earlier this year. What is of public record is that 2,500—actually it's more like 3,000 now—cases are filed each year. What he says is for every case that's filed there are 25 letters claiming infringement. So the number that go to full litigation is only the tip of the iceberg, and while it looks like a good thing that most cases are settled, what you have here is a lot going on that isn't public visible, not a matter of public record.

Now, a capture at multiple levels, I'm not going to spend as much time on this as it deserves. It's interesting that very few political scientists have written about it in the patent system, but I would refer you to Josh Lerner and Adam Jaffe's book, *Innovation and its Discontents*, which does a very nice job of laying out the capture problem at the first three levels there.

I will add—and I will come back to this at the end—that we have a phenomenon here—this is going on in the current patent reform debate—of industry capture. Now, the patent system gets driven by those industries for whom it is the most important, and they're the ones that are willing to put the resources into the lobbying and

especially the lobbying at the international level, and that's why we have this one-size-fits-all provision in TRIPS, because the pharmaceutical industry wanted it there, they wanted to have product patents for pharmaceuticals in every country that bought into the WTO.

Now, under the Court of Appeals for the Federal Circuit—this is the specialized Federal appellate court that has oversight of patent appeals from District Courts and decisions of the PTO—three developments that have made patents easier to get, more potent, easier to assert—and this is in large measure the central thesis of the Jaffe-Lerner book, that you have patents both becoming easier to get and more powerful at the same time, and that this creates a perfect storm. The easier to get is the lowered standard of inventiveness which I will go into in some detail. This is the subject of a petition for cert in KSR v. Teleflex. More potent in the automatic injunctive relief. This has been accepted by the Supreme Court, the eBay v. Merck Exchange case, as to whether injunctive relief must be granted, or whether it is discretionary, as equitable relief normally is.

Easier to assert, there's a heightened presumption of validity which doesn't seem justified given the concerns about the quality of patents that are coming out of the PTO.

I note that on all of these matters the FTC has recommended reform, but there is nothing in the current reform package. There was in the bill introduced in the House for automatic injunctive relief, but that's been pulled out because of concerns outside of the IT sector.

Now, going back to this question of patents easier to get, we look at the standard of inventiveness. We call it non-obviousness here. They call it inventiveness in Europe. And there's this legal fiction that you can actually distinguish between what's obvious and what's not obvious. So if you look at the studies that have been done and the skew of the commercial value of patents, it's probably that what happened is something like this: there are very few inventions that are really inventive, and there are a lot of inventions that are only marginally inventive, so you get a distribution that looks like this. And in fact, we've gone from a flash of genius standard, as articulated by the Supreme Court prior to the 1952 act, to the person having ordinary skill in the art standard, which is embedded in the 1952 act, to this current low standard for combinations, the suggestion test, which is to say that the Patent Office must find the patent acceptable, and it can only find obviousness if there's the suggestion of motivation in the literature to make this combination.

So essentially the Court of Appeals for the Federal Circuit has turned this obviousness test into a novelty plus test. This is a consequence of institutionalized pressures, the capture phenomenon, pressure for more patents from the patent bar, from the patent office since it went to a fee-funded basis back in 1990, and from the specialized court for sophisticated reasons that I won't get into here unless people ask.

So what happens is as these forces push toward lower standards and greater volume, you get this problem because there is a zone of ambiguity around what is patentable or not, and as this gets pushed along the X axis here, you get a larger and larger number of questionable patents.

Moving on to the one-size-fits-all problem. This is a product of focusing on adjudication, so unlike a conventional regulatory process, you don't have a government agency overseeing what actually happens in an industry. The Patent Office just issues the patents and that's the end of it. Ironically, you have this one-size-fits-all model in an increasingly diversified innovation environment, and part of that is in the varieties of new technologies or non-technologies that the patent system has expanded to embrace.

For example, most conspicuously, it does not distinguish between discrete and complex technologies, and this is the problem the IT sector finds itself in now. And it does not acknowledge alternative means of appropriating returns from innovation, such as copyright, such as complements, first mover advantage of secrecy, means that have been documented in the empirical literature.

So this is what's happened in terms of the expansion of traditional subject matter that's really moved in three different directions: biotech towards basic science, which I'm not going to talk about; software towards logic and math; and business methods, or I call them services here, towards the social science, including management science, liberal professions. Complex technologies already are a bit of a problem, and you can see the software problem as the worst case of complex technologies.

I want to point out that the software problem, as it's often thought of, and the business method problem, are actually radically different, even though it's difficult to distinguish between software and business methods in practice. The software problem is that you have one product that has thousands of possible patents in it. So it's a matter of fine granularity. The business method problem is just the opposite, that one patent can

cover many different products and implementations. It preempts different implementations. And this is a great divergence from the classical model of patents being more or less the same as a single product as they are in pharmaceuticals.

Here's IBM's description of the business method problem: and preempt all solutions to a business problem simply by articulating the problem. The software problem is this extreme functional complexity which actually operates at many different levels. So it can operate at the level of the code in algorithms, not very useful to get patents in that area, but theoretically possible, but it also operates at higher levels on up to program features and business methods. There are strong network effects, the ability to block interoperability, importance of complements that distinguish software because it operates in a very functionally complex environment.

Secondly here, the second bullet, is the ease of producing patentable functions. Anybody can write software. You don't need special equipment. You don't have to manufacture it and so on. And there are these great opportunities for extreme economies of scope and scale, global distribution, et cetera, and this is what enable open source, and open source is kind of the existence proof that software is different.

So what you have is this kind of picture, and this is the problem with independent invention. You have millions of producers, probably hundreds of millions of people all over the world writing software. They are creating complex information products and services with tens of thousands of functions. A lot of this is custom, in-house, contracted and so on, not just in products. And you have hundreds of millions of users. So with the possibility of patents at these many different levels, you have a massive potential for building up liability downstream. Actually, I should mention that

the SPC patent that you're referring to is a very good example of that because that patent was used to basically go after retail websites, downstream users, not competitors at the same level and technology as SPC, but users of technology.

Now, this is a wonderful example of propaganda from the recent European debates. This was anti-software patent propaganda. What it shows is a website that has 20 patented functions built into it, which are basically retail operations over the Web. And so this shows the threat to small users of technology that software patents present.

Who should search under these kinds of conditions? Well, if you got a cost of searching that's roughly 2 to $15,000 per function times thousands of functions, and you add in the uncertainty of unpublished patents, patents that are still in the pipeline, plus when you do this you get exposure to the treble damages that willful infringement could buy, does it make sense to search, and who does it make sense for?

So here's testimony in the FTC hearing, Robert Barr saying that there's too many possible patents to make searching worthwhile. IBM used to search, but they tell me they don't search any more. Here's Texas Instruments, also testifying in the FTC hearing, Texas Instruments a big fan of the patent system, done quite well by it, "We have no idea what's in our own portfolio."

Well, this is going back to the President's Commission on Patent Reform in 1966. It concluded that software patents were not practical because they couldn't readily be examined for novelty utility and non-obviousness. That's basically the same thing; reliable search is not feasible because of the tremendous volume of prior art being generated.

So what are the consequences of all of this? What we see here is the systemic failure of the disclosure function. Patents are supposed to disclose information in exchange for this limited monopoly. Prohibitive cost of litigation drive real costs underground and we don't really have a good idea of what those real costs are because they're not public record:

A bias toward capital intensive development models because only those models can assimilate the transaction costs of the patent system as part of the cost of doing business.

Massive embedded liability in the user base, we just saw.

The highest and best use of individual patents is extortion in the hands of small licensing firms that don't have any need for cross-licenses.

And an inter-industry cross-subsidy, a system that works poorly in one sector, but well in another sector is an industry policy that implicitly cross-subsidizes from one sector to another.

Thank you.

MR. KLEMENS: Thank you.

Next up is Emery Simon.

MR. SIMON: So I don't have PowerPoint. I can't do that. I can't do pictures and talk at the same time. Maybe it's a skill I will acquire some day.

Disclosures, I'm not a patent lawyer. I knew nothing about patents 18 months ago. I've learned a lot about patents recently, mostly out of necessity.

I'm not an academic, so I can't do what Brian just did, which is kind of demonstrate for you what kinds of logic and irrationality and mismatching in the patent system.

I know a little bit about software companies and hardware companies and how they look at the patent system, and really, that's kind of what I'm going to talk to you about today, and I think that was sort of your plan to, to kind of put this in the context of technology companies and how they rely on patents.

So this is a story that starts probably 25 years ago, so should go back 30 years. The general thought in the technology industry was that if you came up with new software, you could copyright it, but you couldn't patent it. So the general practice in the industry was to do a lot of copyrighting, and to do copyrighting you don't need to do anything. Essentially as soon as you write the program, you get copyright protection. You've got certain advantage if you register at the copyright office, but basically you get all your legal protections without having to do anything.

And at the time the notion that somehow patents would be available on software was something that the courts were pretty regularly saying that they were not. That changed in 1981. Starting in 1981 people began to look at ways to get patents on software. From a business perspective, there's a tremendous incentive to try to get patents. Copyright gives you the right to prevent others from using your exact code, pretty close. The patent enables you to prevent others from using the underlying idea. So you can come up with 17 different ways to write a print command for a printer, but if you can figure out some way to patent that, nobody else can do it. So there was enormous incentive to get patents.

Despite that incentive, the software industry was very slow in moving in that direction, and in fact, probably until the mid-1990s most pure software companies had no patent portfolios. So big companies, companies like Oracle or Microsoft, as recently as 10 years ago had virtually no patent portfolios. They had a few patents here and there but very little.

Somewhere about that time, maybe a little bit earlier—and I apologize, Professor Dam, because you worked for IBM—IBM is one of the BSA members, but IBM sort of set off a little bit of a chain reaction here. And about that time, IBM, which had been patenting all along because IBM was fundamentally a hardware company and patents were always available on hard things, and as soon as patents became available on software, IBM, just kind of almost a natural reaction, got patents on software as well, and it built up a pretty good portfolio of patents.

And at some point in the 1990s IBM made a decision that intellectual property was an asset, and that asset was to be used, used constructively, and IBM went about licensing a lot of its patented technology.

The practice generally in the technology industry with respect to patents is not so much to license for cash, "You can use my invention for $12.95." It's much more of a cross-licensing pattern, "So I have a few patents, you have a few patents, and we'll license each other." If you're a software company that has no patent portfolio cross-licensing it's an impossibility because you've got nothing to cross-license.

As a result of the Supreme Court decision, as a result of changes in the marketplace, software companies developed an incentive to take out patents, so that was the first wave of pressure for patenting in the software area.

The second wave of pressure for patenting in the software area came out of the dot-com explosion. So if it's a small company and you've got a great idea, and you want to raise some money and you want to go through the venture capital market to raise money, the first thing they'll ask you is: got any assets, because if you go belly-up, what do I have to liquidate to try to recoup the money I've lent you?

Well, most of these little companies had no assets beyond good ideas, but many of them concluded that you could quickly gather an asset by applying for patents, and if you got a patent granted, that was a property right, and that was a property right that the venture capitalists looked at as an asset that would provide a foundation for their loans. So there was a lot of pressure during the dot-com explosion for a lot of little companies and big companies to go to the patent office and get patents. So two waves of pressure are kind of set off by the availability of software patents generally.

About the same time the Patent Office was getting flooded with all these patents, they kind of had a little bit of a short circuit. And the Patent Office had a lot of trouble figuring out how to filter inventions that merited getting a patent, as opposed to inventions that did not. So unlike a copyright, where as soon as you write the code you get statutory protection, in the patent law you get protection only if the Patent Office determines that your invention meets certain criteria. It's got to be new; it's got to be unobvious. It's got to be useful. You know, it's got to meet certain criteria.

Underlying all those criteria is this notion of newness. It's got to be something that hasn't been done before, and generally the way the patent office does that is it compares the invention that you're applying for today to what is out there and known.

One of the quirks of the software industry through most of its development has been that a lot of software ideas, a lot of stuff embedded in computer programs, was not well documented, was not published, was not the subject of articles, it was just kind of in the code, and in order to figure it out you have to go deep into the code and figure it out, and PTO is not in a position to do that. What the PTO generally does is it looks to previously granted patents or looks to published material to determine what the prior art is.

So the prior art database, the thing that you compare this invention to in the software area, has been, and remains, very poor. As a consequence we got to one of the points that Brian highlighted, which is a lot of questionable patents being granted, a lot of things on which patents were granted which really did not necessarily merit those patents being granted.

So we come to about where we are today, which is there's a very large body of patents that have been granted on software inventions over the course of the last 20 years. Some of those are very good, sound and meritorious inventions, and many of them are not. There is a large body of patents which belong to entities that have gone out of business, a lot of dot-coms. Patents are still enforceable. They're still good patents for their term, but there's nobody there to kind of take account of it.

So we've had creative individuals manufacture business opportunities out of these free-floating patents where they basically go about and buy them up, sometimes for very little, and then try to figure out ways to match up those patents against existing products. So hold that thought for a second.

Brian talked a little bit about the fact that the patent law is a one-size-fits-all law, but not all technology is created the same. He can get a patent on a new way to make a sheet or a pillow or a bed, or how to glue a shoe or how to make a new drug, or any number of things. It's a very broad range of things. What is in some ways not unique certainly, but peculiar and special to technology products is that any given technology product may be subject to sometimes as many as dozens, sometimes as many as hundreds, and on rare occasions as many as thousands of potential patents that could read onto the product.

So Chip Lutten, who testified not long ago, which Brian mentioned, again, works for Apple, a popular product is the iPod. Well, Apple has never actually counted it up. Their sense is that the iPod is subject to probably more than 300 or 400 patents, or potentially as many as 300 or 400 patents could bleed onto it.

So when you've got very complex products, assessing the risk associated with patent infringement with those products becomes a very hard thing to assess, and the products get more complicated all the time. Heterogeneity in the industry becomes more of an issue all the time. Interoperability becomes more of an issue all the time. All of these things are subject to potentially patented inventions.

So you put that together with very complex products, and a lot of free-floating patents in the system and what you've got is a potentially volatile mix. So companies, companies I represent—and I represent the large hardware and software companies, but all companies I think are pretty similar—they're risk averse, and the possibility of patent infringement creates a huge potential risk on two fronts. The first front that it creates risk on is very large potential damages, and for example, one of the

cases that's now much in the press is the RIM case, which owns Blackberry, where some people estimate that the case will actually be settled close to a billion dollars in damages. That's a lot of money. So one risk is potentially very large monetary damages.

The second risk, which for some companies is even more acute, is the possibility from having existing products pulled off the market. Now, a product—and I don't mean to cast aspersions—but a pharmaceutical product, a chemical drug, may have three or four patents agreed onto it, and if you find that an element of it is infringing, it's a fairly straightforward proposition to invent around it and to come up with different ways to do it. When you've got a complex integrated system, some elements may be very easy to change; some elements may be very hard to change. And sometimes when you change an element of an existing product, depending upon where it is in a system, it may end up making obsolete whole segments of installed base that go with it.

So depending upon what you change, your printers, your peripherals, your local networks, a whole variety of things, may become either inoperative or much harder to run.

You put all this together, and what you have from a software industry perspective is a patent law that is fundamentally a sound law. There was a time when software companies hated the patent law and wanted it to go away and did not want software patents to be issued. I think there are some within the industry that still feel that way, but as a real pragmatic matter, they are here and they are here to stay, and everybody deals with the reality, and all software companies now apply for patents.

Given that first reality, you then go to the patent system and you look at it, and how do you make it better? And that's kind of where we are, which is the patent

system is a given; the question is how to make it work better for software companies, hardware companies. And really that comes in three principal sets of issues.

The first set of issues is making sure that only things that deserve patents actually get patents, and there are a number of ways to address that, and there are ways that you can do post-grant opposition, which is cheaper than litigation. There are ways that you can give more status to—there are a variety of things that you can do.

The second way to do it is to make sure patent examiners have the skills and time to actually examine each patent on its merits and make a smart decision. I'm sure one of you guys knows the data, I don't, but patent examiners on average look at a given patent for very, very short periods of time, 18 hours. So one of the ways is to make sure those guys have better skills and more time, and then they can allocate their time in a more rational way. Patent examiners actually work on quotas, so each patent examiner is supposed to issue or review a certain number of patents in a given period of time. So the pressure on them to just kind of go through this stuff is very high.

A third element of all this is making sure that the way that remedies work under our legal system make sense for technology products. Probably the biggest issue sitting out there right now is the eBay case, where at issue in the eBay case is, is a product which is infringing always subject to an injunction, or should there be some kind of a special set of rules when the injunction applies or doesn't apply. Interestingly enough, the way the courts have interpreted the injunctive relief provisions is, despite what the statute says, what the black letter of the law says, they've interpreted it saying injunctions are always automatic except for health emergencies.

Interestingly enough, the Justice Department has now intervened in the Blackberry case saying, health emergency is probably too narrow a reading of the law. It's probably got to be read a little more broadly than that.

And the Supreme Court, in accepting the eBay case last week, signaled very clearly that if one's to think through this problem one more time—and for those people who follow the Court closely—I don't—this is a pretty significant development, that the Court in effect chose to rewrite the question that they were going to consider, to be not a narrow question whether judges should have discretion, but to be the underlying question, whether or not injunctions should be granted.

I'll stop there, and we'll do whatever we do next. Thank you.

MR. KLEMENS: Thank you.

Next up is Ken Dam.

MR. DAM: Good afternoon, everyone. You've heard that patent law has always been unitary, which Brian rather pejoratively called a one-size-fits-all approach, and frankly, I think that's appropriate because the patent law is for the future. We never know what new technologies will be coming along. Copyright law has evolved differently with results that I think are rather unfortunate, but I'll stick with patents right now.

The basic argument against software patents boils down to saying that because the patent system was created without software in mind, intelligent, right-minded people ought to be able to create a special system of protection that takes into account the special nature of software, and to very large extent, that's one of the arguments in this very excellent book that Ben has authored.

Now, this argument has been advanced and rejected for most new technologies over the last 200 years since the Constitution first provided for a patent law. And the reasons are interesting, but I'll take an example. Certainly, the Founders, Thomas Jefferson did not have in mind, when he was drafting the clause, that there was going to be something called biotechnology coming along, and none of the provisions of the unitary patent law are specifically applicable only to biotechnology.

I want to recommend to you a report that is just out from the National Research Council of the National Academies, called "Reaping the Benefits of Genomic and Proteomic Research." People from every walk of life more or less were part of the committee, and the people who reviewed the report. And the report rather strongly supports the application of patents to pharmaceuticals. Now I'm sure that somebody will say, "Well, they're really just drugs, and we know that patents are good for pharmaceutical companies," but I think there's more to it than that.

The literature on this subject is quite considerable, and what that literature calls a system or protection that applies only to a particular field of technology is sui generis protection, and there have been lots of arguments for it. The major area in which we have it, plant patents aside, is the Semiconductor Chip Protection Act of 1994. It protects mask works used in the manufacture of semiconductor chips. It has not been a success. Quite the contrary. Now, one reason, and I think an important reason, is that there was a lot of lobbying in the Congress in getting this provision through, and one of the compromises was that there was no protection where reverse engineering was involved.

Well, the effect of that was to create a de facto safe passage for clever pirates, and so the act hasn't actually been used very much, and nobody is satisfied with it. And this kind of giving with one hand and taking away with another is exactly what we have in many parts of the legislative process in the U.S. Congress, where various interest groups get involved and are opposed to each other, as there would be in this particular area.

Consider the U.S. tax code. It's a monstrosity that grows constantly longer and more complex under the pressure of interest groups, and that's one kind of problem that I would expect to find if we tried to get something special for software through the U.S. Congress, and taking the Semiconductor Act again, we don't make semiconductors the same way we made them when the Act was in front of the Congress, and so it's out of date. The manufacturing processes continue to change, leading to the dilemma that a sui generis patent, we can say in general, is either never up to date or goes the way of the tax code, constantly amended depending on campaign contributions and all of the other aspects of the Hill process that we're familiar with here in Washington.

And I dare say the same thing would happen if we took Ben Klemens' advice in his book to make it that Congress ought to make a decision on exactly how we should protect computer software. So while he has written an excellent, highly informative book, which I really recommend to you—and he, himself could probably draft a decent sui generis statute that would be good at least for five years, he is either being a bit disingenuous when he says Congress has an obligation to decide this

question, or he just hasn't been in Washington long enough to be able to realize what would happen to his graph when it got up on Capitol Hill.

In any event, what he is prescribing is more or less what we have in the copyright area, where each new copyrighted technology, if you want to call it that, gets its own 7 pages, and that's kind of unfortunate because it's only the proponent's protection in a particular narrow category that are really able to dominate the process, and everybody else is kind of left out, or it doesn't really pay them to have an interest. If we were amending a unitary statute, do you believe that all the different interests would be involved and would be heard from?

So I am concerned, obviously, about special interest provisions including those in the copyright statute, and for those of you who are our techies here, I think you all know that academics and most software people are unhappy with the Digital Millennium Copyright Act that bars circumvention of technological locks preventing copying. That's a good example of complex legislation designed to give something to each of the interests involved in the controversy over protection on the one side and copying and fair use on the other. This is what I would expect to come out of an effort to get a sui generis statute for software. So that's the negative side of my review of the book.

The positive side is that I totally agree that we need action, but we just don't congressional action. The present situation in the software patenting arena is highly unsatisfactory for the reasons that the two previous speakers mentioned, and I will review from the standpoint of the way I look at the problem. The problem lies in the failure of the courts and the Patent Office to interpret the patent statute according to

its terms, and especially according to traditional standards that were used in the past in prior new technologies.

Now, first of all, the patent statute is really fairly simple. It says, in essence, innovation, in order to deserve a patent, has to be novel, non-obvious and useful. There are some other provisions, but that's the essence of the patent statute.

It's rather technical, the law, particularly as developed by the court, is rather technical, but it's also rather abstract because it has to cover technologies that we can't even foresee today. That's the idea of the unitary statute; you want something that will be valuable in the future. But though technical and abstract, the law is very sensible it seems to me. For example, to know what is an obvious innovation, we ask what would be obvious to a person skilled in the art? You can take testimony on that, and that's something you can know. In other words, what would Ben Klemens find obvious in the case of the software art? That should be the standard.

There's a big controversy in the literature about whether the Patent Office has been too strict or too liberal in this area of the non-obviousness criterion. I suspect Ben would say that they've been too liberal, and frankly, I think he's absolutely right on that. At the same time there are a few legal issues involved in the obviousness area, and one that may be heard by the Supreme Court in the KSR case, which has already been mentioned, is whether it's okay for the Patent Office to combine, quote, "references." What is a reference? Well, a reference is what has been out there in the past. So, for example, if you have a complex innovation—and we've heard how complex software is—and each part of the innovation is obvious in the light of the prior art, that is to say, the light of an existing innovation. Then the question is whether you can combine those

prior parts, which are all obvious, to find that the complex innovation as a whole is obvious.

Now, I think the answer to his is very simple, and it's dictated by general principles that we've always lived by in the past. The question is whether it would be obvious to a person skilled in the art like then to combine the three parts. That should be a factual question, not a legal question, although the legal question has been—legal answer in the past has been that there has to be some suggestion in the prior art that you should combine these references before you can actually combine them. That's the black-letter law on the subject. I think that's wrong. It should be just a question for a person skilled in the art.

Now, another set of issues involved disclosure. This issue involves what the patent applicant has to disclose in the patent application, which after all, becomes public knowledge 18 months after filing, and becomes a public document available to everyone. The Patent Office and the courts have been fearful that if a lot of disclosure were allowed in the software area, it would lead to piracy. So they have allowed special rules to come forth in the description of a software innovation and in the so-called enablement requirement, which is that you have to, as an applicant, say how to make and use this particular innovation.

The result is that today patents are issued that exclude new entrants to a field. That's what a patent is; the power to exclude others, where all that is really claimed by the patentee is that he has some new method, some unspecified new method for solving a recognized problem. Now, not only is that bad because it leads to too many

patents where there is no real innovation, but it is also bad when there really is a major innovation involved.

And why is that? Because disclosure is very valuable. It is desirable because it helps technology to progress more rapidly by adding to the general level of knowledge, as well as by stimulating competition through inventing around research, and by stimulating research to improve an already patented technology. So the lack of disclosure we have in software I think actually impedes the pace of technological progress.

But perhaps the biggest problem lies in the business method patents, which you've all heard about. Business method patents refer to patents issue, where what is claimed is not a technological innovation, but rather just a new method of carrying out a business function.

First, a number of these patents was held back by requiring that the new business method had to be implemented through technology, in other words, by a computer and therefore by software, general purpose computer, for example. But now in a case known as the Lundgren case, the Board of Patent Appeals in the Patent Office has held that business methods do not even have to involve software because there is "no technological arts," in quotes, requirement in the patent law. This question is now before the Supreme Court in the Metabolite case, which is not actually a software case, and not even a business method case, but the Supreme Court has indicated what they want to hear argument on, which is this issue about whether you can just say, "Here's how you put your pants on in the morning, and I've got a new way of doing it, and therefore I get a patent," or something like that. "Here's a new tennis stroke."

Another big problem involves the competence of the Patent Office, which Emery has alluded to. It's not that patent examiners are not competent, although I do think that many of them are not really up to the task, and I happen to feel that way because I've had a few in my courses. The problem is that the Patent Office has no way of knowing what is really new in the software area. That was particularly a problem some years ago, but still a problem now. They don't have access to all of the prior art. And most feel the prior art is all in the prior patents, the prior patent applications. But since software patents are relatively new, as you've heard, most of the prior art is just in the heads of software engineers and in the software that may not even be currently used or maybe is only described in some technical paper which is not generally accessible to the public.

The Patent Office and some companies working jointly have tried hard to overcome this problem, but it's still there. One way to solve it would be to introduce an opposition procedure in the Patent Office so that outsiders could come and tell the office what the prior art really is. Today that opportunity comes, if at all, only later when you have this high-priced litigation process that you've heard about. In Europe there is such an opposition, an administrative opposition procedure, and so the United States is lagging in this area.

There are a number of other problems out there. You've heard about the problem of injunctions, and I think this is related to the problem of our legal litigation process. If you have a very complex program and just one or two patents are infringed by that program, the possibility is that the patentee can close down the whole program, and therefore, the whole business of, say, research in motion, and the Blackberry.

One obvious answer would be to say that in that situation the patentee is only entitled to damages with regard to his own narrow patent, but not entitled to an injunction. Now, many people don't like that because it sounds like compulsory licensing, which the United States has always opposed. However, the question is now before the Supreme Court and so we may get some information on this, some guidance on it. However, at the end of the day the fundamental problem is our own distinctively American system of high stakes, take-no-prisoners litigation, with, you know, massive discovery of all your old e-mails and so forth, where the patentees can wage all out warfare against any true software innovator, especially a successful one, and continue the fight for years. Naturally, the successful software company finds it simpler to settle for money than to fight, especially if it finds itself suddenly out of business because of an injunction.

And interestingly enough—I haven't done any study of this—but I believe there must be in many cases where the settlement amount is actually more than would be awarded in damages at the end of an infringement action because they want to continue their business, and not being able to continue their business, are willing to pay more than they might actually have to pay at the end of a long fight over damages. So that is a really difficult problem.

It's a big problem, but notice it's not really a software patent issue. It's an issue about our whole system of litigation, including the role of injunctions, and it rises in many fields of substantive law, not just in the business software area.

Thank you very much.

MR. KLEMENS: Thank you, Ken.

Next up we have Dan Ravicher. I don't know how long this will take, so maybe you can circulate by stepping over to the bookstore.

MR. RAVICHER: It won't take that long.

[Brief recess.]

MR. RAVICHER: So let me ask before I start, how many people here have a strong opinion about software patents? I don't want to know what that opinion is, but how many consider that they have a strong opinion?

[Show of hands.]

MR. RAVICHER: So very few, that's good, about half of you. I just like to know that, because I like to know how many people are here that find out new ways to advance their opinion or actually help shape their own opinion.

Since it's Christmastime, I like the movie "Scrooge," so we're going to go back to software patents past, and software patents present, and then we're going to think about software patents future, and then we'll talk about various issues that some of the other speakers have already raised, and I'll just maybe give it a little bit more granular, and maybe a different perspective.

When you ask a lot of people when software patents were first allowed in the United States, they will cite this 1998 case called State Street Bank, but if you ask anyone knowledgeable of the patent system, that would actually be a wrong answer if you were to put it down on an exam question. Software patents—and there's an issue as to how you define software patents, but generally recognized by patent attorneys—were allowed, as Emery said earlier, back in the early '80s. But that doesn't mean that State

Street Bank from '98 didn't have a significant impact on the software patent system. So let's look.

Brian showed this graph earlier. It had the little pink line along the bottom, which I have now increased here. So what you see is a general, you know, pretty average rising of software patents in the '80s and early '90s, which could mirror any kind of technological development I would expect.

Then in 1998—up until this time the Patent Office had been holding a somewhat fuzzy line on not allowing patents to issue on pure software. The '81 case, which is Diehr—not "deer" as in in your headlights, but Diehr as in D-i-e-h-r, in case you want to Google it and read it later—said so long as the software was implemented in some hardware fashion, as long as it was some part of a machine, that was patent eligible, but if it was just individual by itself, that wouldn't be patent eligible. So a savvy patent attorney said, "Okay, instead of getting a patent on software, I'll get a patent on a machine, wink, wink, that does something that other machines in the past weren't capable of doing," the machine itself not being the novel of non-obvious aspect of that patent claim, it's the software on the machine, but that's the way to get over the Diehr hurdle that the Supreme Court had put in front of patent attorneys.

The Patent Office did a good job of holding that line, but there were still lots of people who had applied for these patents, and they were pending, and the Patent Office would keep rejecting them for what's called Section 101 grounds, which is non-patent eligibility.

Then in '98, the Court says, "You know what? That's kind of a fuzzy line, doesn't make much sense. "It's kind of silly to require people to just add these unnovel,

ridiculous hardware elements to claims. Pure software is fine." So what you see then is an explosion.

[End of side A, begin side B.]

— backlog that the Patent Office have kind of been holding back, right? They've been holding back the water, some was soaking through, but they're holding back most of the water. Then the courts said, Patent Office, you're doing the wrong thing, let everything through.

So this was a bunch of clearing out the backlog, and you see it's very steady here. Right? Notice that. For about two or three years it's almost flat, and that's because the average pendency of a patent application is two to three years. So two things happened in '98: A, the Patent Office released its backlog, which it continued to do over the course of the next couple years, and people started applying for more pure software patents. And so you see the evidence of those applications occurring three years out from the decision to where now you see the traditional slope again.

So this is basically the pendency period of software patents that was being allowed, and then this is the result of the increased behavior that actually happened here, but the statistics don't show it until the patent actually gets issued.

So some rough, off-the-cuff guesstimations. How many people have an idea of how many software patents—and, again, there's a definitional question, but, generally speaking, how many software patents are issued by the Patent Office every single week? Any rough guess? And if you looked at the numbers on the chart, you should be able to do a rough calculation. Anybody want to guess?

MR. : Five thousand.

MR. : Five thousand? That's actually—no. Six hundred per week. But if you multiply that times 50 weeks, you're talking a pretty big number.

How many patent lawsuits are filed every week? And Brian already hinted at this. That's about 55 [inaudible]. Now, what percentage of those lawsuits are software-related? A small percentage, so they're not all software patents. But we still have an increase in patent litigation. And the cost of defending oneself from a patent lawsuit, as Brian indicated the statistics show, from a year or two ago, I think that data came out, $2 to $4 million for an average size patent case.

Now, one of the problems with software patents is not just the lawsuits. A lot of people think, well, the only thing you can do with a software patent is sue someone in court. False. Wrong thing to do. Emery already told you a very valuable thing, perhaps one of the most valuable things, that a lot of companies do with their software patents is get a cross-license. They never sue anyone. They just negotiate a cross-license.

What are some other valuable things you can do with your patent? Well, you can threaten to sue people. There's value in the threat separate and different from the actual following through on the threat. Well, how much does it cost to threaten someone with a patent lawsuit?

[Inaudible comment.]

MR. : Exactly: Thirty-seven cents.

[Laughter.]

MR. : Okay? That's how much it costs to threaten one with a lawsuit. Once you've been threatened with a lawsuit, patent law requires you to comply

with a duty of good faith to avoid infringing the patent. That duty of good faith, although it's been somewhat modified by the court recently, for all effective purposes requires you to go out and hire a reputable patent attorney. My card will be available—well, actually I don't do private work. I have some friends. And the patent attorney, reputable patent attorney, has to write you a letter on their letterhead that says their opinion as a patent attorney is that you don't infringe.

Absent that letter, the courts are very likely to find that you did not comply with your duty to avoid infringement, and, thus, they will increase the damages awarded against you in court, which has the near-term effect of increasing your exposure. Right? So at the end of the day you may be liable for more damages because you didn't comply with your duty, which means you're a juicier target for people who would like to use this threat.

And how much does such an opinion letter cost? Well, we do have to put our kids through Harvard and keep our house in the Hamptons, so they're not cheap. So it costs 37 cents to threaten someone, and it costs $30,000 to $50,000 to deal with a threat once you're [inaudible].

Now, is this good or bad? I've tried to keep out my editorialism. I won't hide the ball on you. I'm opposed to software patents economically, technologically, and socially. But what do other people think? I don't run the world. Well, large software companies, it depends, they only have one voice. Some large software companies are opposed to specific software patents or specific flaws with the software patent system. Some software companies are opposed to software patents in general. It depends. Small software companies—now, this is actually—companies that actually produce software.

It depends whether or not they've been threatened. Small software companies I talk to who have yet to be threatened or sued like patents because they believe if they get themselves guns—which I call patents "guns," for better or for worse—if they get themselves guns, they'll be better protected. Right? Who wants to walk down a dangerous street without your own guns?

But they're don't realize they're in an ecosystem that gives everyone else a lot more guns, and the number of guns you have is directly related to how much money you have. So when you're relatively weaker financially than your competitors, they're going to be able to get more guns. They're going to be able to outgun you. So it's actually not helpful to have a system that allows guns to be produced.

If you ask patent attorneys, it's great, because we make lots of money—right?—so we love the system. And the more patents, the better; the more issues to fight about, the better. When I teach law students, every time they ask me a question that doesn't have a yes or no answer, I call that a "ka-ching." Does anyone remember the Wendy's commercials with the guy in the drive-thru, and he was just going, "Ka-ching, ka-ching, ka-ching"? Because the patent attorneys issues ka-ching, any issue is a ka-ching, and more patents are more ka-ching. So it's all gravy.

Now, lawmakers are very similarly related to patent attorneys because the larger the system is, the more power they have, the more impact they have, the more important they are, the more people want to talk to them about it. So they naturally trend toward advancing larger patent systems, and Professor Dam made an excellent point about legislation which helps a small group of people and hurts a distributed class of people is the easiest type of legislation to get passed, because those who benefit from it

will be there knocking on people's doors and having lunch, and those who are harmed by it may not even know what's going on. So it's very easy to get passed, and that's basically what the patent system is. It benefits those who get patents and it harms the rest of the public.

Now, patent trolls, they would claim that they are a small software company, so I distinguish them from legitimate small software companies by saying they're just a non-producing entity. They're a company that holds patents and researches them, but that's their only business activity or substantially their only business activity. Well, of course, they're in favor of patents, right? Their business is made off of threatening and following through with threats on patent suits. So they're in favor of it.

And the non-patent holding public, well, we don't care. Right? My mom doesn't even know what a patent is, much less care about—well, my mom does, but most of our moms don't. And so due to an unawareness because the patent system is so opaque and the effects on people are so not transparent, most of the public doesn't care.

So there is—you know, you were bait-and-switched a little. There was the marketing that this would be a debate. I didn't hear very many issues on which we would have hard-core, fiery debate on. Most of us agree that there is some reform needed, and these are all issues that they've already talked about. But my question at the end is: Is the system going to get better? Should we just hang in there—right? Technology is growing at a lightning-speed pace and we're going around the corner and we just need to hang in there and once we get around the bend, we'll straighten out and we'll fly right and everything will be okay. Let's just give the Patent Office some time to

catch up with the technology. Let's give the courts some time to catch up with it. Let's just stick with it, and we'll be better off at the end.

Or should we just recognize this is a failed enterprise, this is just something that we've got incredible advance of software without software patents. Since software patents have come into the world—I'm not saying there's a correlation but at least randomly, we had the dot-com bust at about the same time as all those new software patents were coming out. So, you know, we have to ask ourselves is this just a systemic flaw that we should recognize, and I draw an analogy—not too close an analogy, but somewhat of an analogy. How many people have heard of Governor Ryan from Illinois? Right? And so Governor Ryan ran on a very strong pro-death penalty platform. People asked him when he was running for Governor, "Would you ever commute anyone's sentence on death row?" "Absolutely not. I'm against criminals. The death penalty is good." Right?

Then through his experience as Governor, he saw how systematically flawed the death penalty system was and how it did not give people a fair day in court. And at the end of the day, the people being sentenced to death he could not absolutely in his conscience know that was the right decision. On his last day in office, he commuted every single person on death row to just life in prison. And so we need to ask ourselves, Is there a similar kind of awakening that may be happening here in the software patent world that it's not just isolated incidents of harm, it's not just that there's poor patent quality and patents are too broad and all these other things that are wrong with the software patent system? Is there actually an overarching problem that just encourages us

to just say, you know what, we were fine without it, we don't need it now, let's move forward.

One thing I bring up when I bring that up is, well, let's have a world where people can choose. Right? Whoever wants to stay in the world with software patents can choose that world, and whoever wants to stay in a world without software patents can go to that world. So it's kind of like, well, you don't get any, but you can't be harmed by any.

Well, we kind of have that world already because let's ask what large corporations do. Do large corporations sue each other for patent infringement all the time? Are IBM and Microsoft suing each other all the time, or Oracle, or Time Warner? No, because they pretty much recognize it's better for them to opt out of the patent system in effect by signing cross-license agreements than it is to actually take all their guns and shoot them at each other and see who's standing at the end of the day. So they implicitly recognize there's a benefit to opting out of the system, which perhaps the rest of us could also have the same benefits from.

Now, reform is difficult because there are other interests; namely, people that benefit from the patent system and, you know, these were already talked about earlier. So what about the future? These are the thoughts that I'd perhaps like to leave you with.

The first is keep your eye on the ball. What is the goal of the patent system? A lot of people would say it's to encourage investment in research and development. I think that's wrong. A lot of people think it's to reward patent holders. I say that's wrong.

To me the goal of the patent system is to advance technology that is made available to the public, and so there is a point along that line at which, yes, encouraging investment does advance that end. Rewarding inventors, yes, can advance that end. But you can overcompensate in those areas, which then has a negative effect on advancing technology available to the public. And so anything that does not advance the technology that you, me, your mom, your grandma have access to is harmful and not the goal of the patent system.

Keeping your eye on the road means we have to be cognizant of where the reality is. There are speed bumps that come along the way. Let's not get distracted by individual isolated cases. When I see the press articles about, you know, the swing-sideways patent and how silly that is, I think that does a disservice to our discourse about this topic because there are more important issues going on here than just whether or not some 13-year-old got a patent on a sideways swing.

The Supreme Court cases, you know, the patent system has been characterized in our history as being very much one of mood swings. There was a very strong anti-patent mood in our courts many decades ago, and the response to that was a mood swing over to the opposite end and the creation of the federal circuit and modifications of patent law in Congress. And now there's a bit of a mood swing back, and maybe we don't want to be mood swinging so much as trying to plot out a direct course straight ahead.

Keep your eye on your wallet is my point to the general public to say think about how the patent system is affecting your daily life. Are things more expensive or cheaper? Are better technologies available or less available? How is the

patent system affecting your life? Think about that, realize that, and then express your opinion.

And keeping your eye on the prize. How many people remember these? I was shown these when I was in elementary school about the civil rights movement in the '60s. There was a whole PBS special. Keep your eye on the prize means our laws should do good, right? Our laws should do beneficial things. Sometimes in the patent system we all get so focused in on, you know, should this patent be allowed or should it—I think it's good to step back and say, Is this benefiting society?

You know, software has been held by the courts to be a form of expression protected by the First Amendment. Should we take pause and think really do we want the government to give private corporations the right to regulate speech?

Now, if you disagree with the courts that software is speech, then take that up with the courts that have held that. But I believe that software is speech and it is protected by the First Amendment. So should we have patents issued that infringe on our First Amendment rights? I have some concerns about that. I have concerns about access to technology. I have concerns about access to the patent system itself. So just stepping back and seeing the more global issues that the patent system affects I can help us make sure that we merit out that direct court. That would be best for all of us.

Thank you for your time. Thank you, Ben, for coordinating this panel and letting me attend at the last minute, and I think now we're going to have questions.

MR. KLEMENS: I guess before we start with questions, if any of the panelists want to make comments [inaudible].

MR. : I think the only person I really disagree with is Ken. I'm not sure [inaudible]. My sense is that you work for IBM, who has changed its position radically on software patents over the years and [inaudible] swinging back to a position that's now closer to what it was 30 years ago, 35 years ago, when James Birkenstock sat on that President's commission and argued very forcibly against patents [inaudible]. And IBM today finds itself in this kind of situation that Emery was describing where it's not suing anybody for—it has done very well with its licensing of patents, but they tell me that the freedom of action that it gives them is worth $8 billion a year as opposed to the $1 billion a year they get from licensing their patent portfolio.

Now, if that's the case, what we have with cross-licensing is a very different kind of patent system than what legislatures have in mind. It's a patent system where the name of the game is not the patent but the portfolio. And the reason the software industry bought in over the long run to software patents is—and back in '94 when we had the hearings on software patents, every pure software company was against them with the exception of Microsoft. The acceptance that I think we see nowadays is because in a portfolio world, once you've got your portfolio, you are better protected against your competitors than you are if it's a patent-oriented world. That portfolio creates a barrier that can be evergreen. You keep adding to that, and you make the cost of entry for the new guy who doesn't have any patents, who has to come into these [inaudible] arrangements by paying side payments, you make entry very difficult.

At the same time, I think both the hardware companies and software companies and IT got oversold on patents by their own patent departments. And so what you see now with the troll phenomenon is the fact that lowering standards of patents,

which helps you build portfolios, also helped the troller. And they've, in fact, created this troll phenomenon by lowering the standards and making patents easy to get.

MR. : Well, I suppose I'm supposed to comment at this point. I left IBM in 1992, and I know—I can think of only one occasion where I even talked to them about this issue, and that was in 1994-95. So I don't really know what they're thinking, although I did do a little research and found that Mr. Kappas (ph), I think is his name, has been taking a position which raises a lot of questions about a lot of the software patents out there. I don't think he's saying he's opposed to software patents, I don't believe.

What actually happened shortly after I left was that we had a—this was a—the patent lawyers and the patent licensing people and the standards people and so forth were corporate staffs which reported to me. And so I wasn't involved in the day-to-day work, but there was an effort shortly before I left to decide what was our real interest in the area of patents—not software patents as opposed to hardware patents, but patents. And prior to that time, we had always—we, I'm now talking about IBM, had always cross-licensed, but almost all of the patents were hardware patents, and there were a limited number of hardware companies. They began to grow rapidly around 1990, and there it was remarkable. The cross-licensing was needed because there were patents and you didn't want to have your product held up. You wanted freedom of action.

And it was interesting. They were able to figure out how much the patents were worth by using technicians to do it, and there were cross-payments. And

IBM was always positive in the balance of payments; a lot of small companies were always negative. But nobody really wanted to drop out of the system.

Now, with software patents, they got a lot more complicated, and even with some of the hardware companies. I remember Motorola and Texas Instruments, they weren't interested in cross-licensing or patents, and that's when the litigation really began. But then, as software developed in the '90s and thousands of new companies entered the field, it was a whole new ball game.

Meanwhile, IBM, I think they now dropped to the bottom line, about $1.6 billion every year out of patent royalties and these cross-payments, which IBM had a positive balance of payments, let's say, and that's one of the reasons that they like it. I don't know that they have an ideological position on whether patents are good or bad. It's strictly a business proposition.

Now, IBM has done some other things over the years which are quite interesting. Where they have gotten patents in areas that—just by accident where IBM is not in the business, they basically auction the patents to the person who will pay the most in terms of royalties. Some of them for public relations purposes, where it involves health or basic science, they dedicate to the public. But in many, many areas they just publish, because once you publish, showing that IBM was first, then nobody else can get a patent. So that helps IBM have freedom of action, and I think that's one of the most interesting factors. And I would be interested in knowing whether IBM does more of that or less of that, because that is an alternative for that large segment of the software industry which really doesn't believe in patents. They ought to publish what they have. Then they'll really make it open source, by the way. Everybody will know about it.

MR. KLEMENS: Any other comments from the—so let me start with a few questions that I have.

The first comes up with NTP v. RIM. This is the Blackberry case which has been all over the news for the last few weeks. One interesting feature of the case is that our Blackberries' RIM servers are actually in Canada, so the software, which is infringing on these patents, much of it is actually in Canada. However, "control and beneficial use" of the infringing system was within the United States, so the Court of Appeals for the Federal Circuit said that this software is still infringing U.S. patents. What—

MR. : I just want to make a clarification. That patent actually had two types of claims. It had method claims, method of delivering e-mail to a portable device; and it had apparatus claims. And the Federal Circuit said since the server is in Canada, it cannot infringe the apparatus claims because the entire apparatus, the entire structure is not here. But the method claims, you're correct, they said those are infringed. So they did make—just to be fair to the opinion, they did make a distinguishing point there, which I don't think is correct, but I just wanted to make sure that the opinion the Federal Circuit handed down was—that everyone understood the difference. Everybody gets the difference between an apparatus claim and a process claim or a method claim.

MR. KLEMENS: Thank you.

So more generally, my question to the panel is: Is the U.S. regime—the U.S. regime differs very much from the rest of the world because the U.S., arguably, has the strongest software patent regime. Is this creating conflict between the U.S. and the

rest of the world? And how can this be resolved? How can patent law be harmonized, given that any piece of software can be anywhere on Earth?

MR. : I think that's the wrong question, which is it's not so much a question of U.S. patent law being out of step with other countries—software patents are available everywhere. They're not acquired with the same regularity as they're acquired in the U.S.

I think the more fundamental issue is that we just have a different approach to how we think about rights and litigation, and I think that's where the problem is. I don't think it was being more or less lax in granting software patents. [inaudible] our Patent Office has, in granting software patents, [inaudible] true for every Patent Office.

MR. : Let me comment on that. [inaudible] situation in Europe where the European Patent Office is not quite as liberal as we are in granting them, but it does grant a lot of software patents. But that doesn't mean [inaudible] because you have to take those patents to the national court to [inaudible]. That's a lot more questionable than—and particularly there have been a number of cases coming out of the U.K. recently which hold that the technical contribution, which is the main distinction, the main operational distinction between the standards in the U.S. and in Europe [inaudible] software, pure software patents, is higher than the practice of the EPO [inaudible].

MR. : Just to make a small law professor-type point, obviously patent harmonization is desirable, and it's often been attempted and sometimes it's successful. But here there are two problems. We have a different system of litigation, as I tried to suggest, and I think it's gone way too far, especially where you can get

contingent fees and all that kind of stuff, things that have been considered malpractice or worse in many countries. But, you know, this is not a problem special to patent law, much less especially to just software patent law. It is in practically every field.

You all know about Sarbanes-Oxley. Well, the world is so inter-penetrated, firms operate all over the world; European firms are really up in arms about all of the requirements that the Sarbanes-Oxley law imposed on corporations a few years ago because of the various corporate standards.

So, you know, I think it's very—people get too specialized, and I'm not merely disagreeing with a lot of the things that are being said here. It's just that I think you need to realize that a lot of things that seem to be impacting the software industry are not software industry specific. They have to do with a lot of other things, and the whole—patents are territorial, but so is a whole lot of legislation in many, many business fields. So we got to keep our eye on the ball about what the problem really is, whether it's patent law, litigation, the lack of patent harmonization in the world, or what exactly the problem is.

MR. : Looking at this response from the argument you made in the book, the real [inaudible] here is that at the policy level. And what we've just seen in Europe is a protracted, very vociferous debate on software patents, of which the antagonists ended up receiving the European Voice of the Year Award in two categories. And although what happened this summer was inconclusive, that is to say, [inaudible] basically shot down, this is something that had never happened before on second reading. Parliament totally rejected what had been done by the Commission [inaudible]. Still, there's a very well-defined struggle going on over whether software should be

patentable or not, and this is not an argument that there should be a sui generis regime of software. That's not in the cards. That's not part of the [inaudible]—

MR. : Thank God.

MR. : Yeah. So it's just a question of whether it's copy—

MR. : There's [inaudible].

MR. : [inaudible]. It's not an issue there, so it's [inaudible]. Is copyright alone sufficient or do you need patents as well? And one way to look at this as a matter of policy is why do we have this regime that 20, 25 years after it is initiated, is still as controversial as ever? Do we want laws that are this controversial? I mean, look at copyrights by comparison. Copyright worked so well between competitors that it was a non-issue. Everybody knows that it's not proper to take your competitor's [inaudible].

MR. : But people still do it.

MR. : People still do it.

MR. : I agree with Ben. I do have concerns about our courts not respecting our borders and, more importantly, not respecting the borders of our neighbor countries. I mean, there's another bad case as well where a very large software company sent one disk with one copy of a program overseas. It left the United States, outside the United States, absolutely without question. Then once it was outside the United States, several copies of this master disk were then made to put on computers, and the Federal Circuit said, Well, even though those acts were outside the United States, we are going to calculate infringement based on all those copies made outside the United States because when you sent the master disk to them, you knew that it would have that effect.

I mean, these kind of decisions, which in my mind express an arrogance about our patent system being the absolutely best one and right one for everyone in the world, really caused concern for me, and I think it should cause concern for most people. So I agree with that spirit of your question.

MR. KLEMENS: So I guess we should open it up to the floor. I've got a long list of additional questions, but if anybody would like to—go ahead. If you could state your name and—

MR. MITCHELL: John Mitchell. I wanted to maybe tie together comments from Dan and Brian to form a question out of that. From the standpoint of defensive patents, anecdotally it seems that my friends who are patenting these days are saying, "I don't plan to exploit it, I just don't want someone else to come along and tell me I can't continue to exercise my invention because I failed to get a patent, and I'm being encouraged now, not in an effort to keep a competitor out, just to be able to keep a competitor from keeping me out." And so I guess a premise question is whether in any of your data you see that there's that kind of defensive patent. And if so, turning around to the free speech aspect, I think it's absolutely right that there's that element of free speech, but in addition to the question of, you know, whether we need fair-use rights or things of this nature, whether if I am, in fact, feeling obligated to get a patent at some expense just to be able to keep on speaking, whether that itself is not, in effect, a burden on speech through the system. And maybe before Dan comments on that, we should find out whether the premise is correct.

MR. : Well, this is what Ken was talking about, defensive disclosure as a way to keep that from happening without going to the expense of [inaudible].

MR. : I'm a little more concerned that you are mixing together two situations, and please clarify. One would be where a firm is really the first, but it doesn't patent; and, therefore, it finds out that was a mistake; so now, whenever it comes up with something really original, it patents it. Or as opposed to amassing a war chest of patents so that you can fight whoever comes along—

MR. : No, the—

MR. : Those are somewhat different—

MR. : Right. The first situation was what I was looking at, a situation where a firm doesn't really see necessarily an economic benefit directly in exploiting the patent, but sees a real economic downside if someone else comes along and prevents them from exploiting it even though they were first.

MR. : That's where the public disclosure works. However, some firms wouldn't want to disclose because they might be inviting litigation, so they'd just like to keep it secret as long as possible. And trade secrets are one way to protect inventions, at least for a certain period of time.

MR. : But don't get misled into thinking just because you have patents that cover your inventions other people can't have patents that stop you from doing that. Like I said, a patent is a gun, so it gives you the right to try to shoot at people and defend yourself, and the more guns [inaudible] defend yourself. That doesn't preclude other people from having guns pointed at you, especially in the software field,

where, like Brian has said, you know, your one invention, your one product, or your one service could have several patented inventions in it. I guarantee you RIM has tons of patents. That doesn't stop NTP from being able to use its five patents to shut them down.

MR. : Has the First Amendment issue been argued in any case that you're aware of?

MR. : I've never seen it. It's one that I think needs to be raised. We do have respect for constitutional rights and other forms of laws that regulate ideas, in copyrights, fair use that you use identified, trade secrets, and trademarks. But there's no absolute discussion I've ever seen about the—in law about the need to respect our civil rights under patent law. There have been some academic law review papers on that topic, and it's a burgeoning topic which I think is going to be discussed more as we go forward.

MR. : Is it possibly an issue [inaudible]?

MR. : It could be an issue the Metabolite case. The Metabolite case is basically a company that sells tests to measure your blood level of a certain thing, and then there [inaudible] medical literature that if you have a high level of this thing, then you're sick with a certain disease named "thing" disease. So they got a patent on several things related to their testing of your blood levels to see if you have the thing, but they also have a claim that says what infringes our patent is anyone who knows your level of thing and then diagnoses whether or not you have thing disease. And so the Supreme Court has asked whether or not that is a claim that should be eligible for patent protection.

And so the First Amendment issue there is do we want to have claims that effectively preclude all speech regarding thing disease and its correlation to the presence of thing in your body.

MR. BAND: I'm Jonathan Band. It seems—I agree with the notion that having sui generis exclusion for software patents would be kind of pointless because clever patent attorneys will always find a way to write the patent so that it doesn't read on something that looks like software, but it would always include something else and thereby be able to avoid any kind of statutory exception.

At the same time, all the panelists seem to agree that there are problems with software patents, and some have suggested, well, it's mainly the courts and the PTO, not the underlying patent statute. But it seems that the courts and the PTO over 20 years have not been able to fix the problem, if anything have exacerbated the problem. So doesn't it seem that statutory reform is the only way at the end of the day to get the courts and the PTO to apply the Patent Act correctly?

MR. : [inaudible] only way, but it's not going to work. We've got software stuck in the system that's dysfunctional at the political level as well, because now we're locked into this one-size-fits-all model and we can't get out of it.

MR. : And when you say "the courts," you have to realize that with the creation of the Federal Circuit, what Congress did is pretty much eliminate the second level of appellate review for patent cases. A lot of people think they just funneled it to the Federal Circuit. What they in fact did is take out that second level of review and say all appeals go to this one Federal Circuit. And so the Supreme Court saw it and said, well, we don't need to ever take a patent case again, right? Because the

Supreme Court is only here to settle differences amongst the circuit, between the Ninth Circuit and the Second Circuit. Well, if all patent cases go to the Federal Circuit, we don't need to get involved, because they'll never disagree [inaudible].

Well, now the Supreme Court this term is being highly active and saying, wait a minute, they're actually doing a lot of things that may be bad policy or may not be in line with the congressional statute. So the Supreme Court has been dormant in this area for a long time, taking maybe one patent case a year and not being very clear about the decision. This term they're being very active, and so we'll have to see what effect that has on it, in addition to the legislative effects—or efforts that are underway, you know, this term, and I assume next term as well.

MR. : I think [inaudible] is really hard because we somehow—we are looking at the patent system in a narrow prism right now in the software industry. But that patent system works for lots and lots and lots of different industries, and I'm confident that if we were to look at it through a narrow prism to how it applies to mechanical inventions on new engines or how it applies to airplanes or how it applies to any given sector, you'd come up with spots and blemishes like we have found or that we find that apply to the software industry. There will probably be different spots and blemishes, and you have to be pretty careful in how you change a one-size-fits-all law.

Having said that, I'm not sure—this is me talking, not on behalf of any of the people that I represent—I'm not sure that a one-size-fits-all law really makes sense in today's world for the patent law. Technology has become too complex, inventions have become too diverse. That's not to say it's a sui generis law. But just like in the contract law, we have a series of differences based upon whether a broadcast or a public

performance or various different exercises of the right. It's probably worth thinking about similar concepts being applied in the patent area as well.

MR. : There is an example of that, and there is a provision, I believe in the obviousness provision, for a special [inaudible], and I don't know whether that's worked well or poorly. I just haven't looked at it, haven't been involved in it. If there are any patent lawyers around who are familiar with that, was that special interest legislation just to give biotech firms more power or what was it?

MR. : [inaudible] and they tried [inaudible] and there was too much resistance from others who didn't want it [inaudible] consequences, and they ended up settling for this special carveout.

MR. : It was limited, a limited carveout. But to the extent it was really just to strengthen the industry, not to weaken it—right?—in terms of—

MR. : Yeah.

MR. : Well, that's the special interest aspect of this. I don't know if we had a shootout in the United States Congress such as they had in the European Parliament—which is not really a legislature, it's kind of a fail-safe, to pretend that they really have democracy in the European Union. Whether—what would actually happen if we had a big shootout in the U.S. Congress on software patents? I'm not sure that the techies in the audience would be all that happy.

MR. : [inaudible], which is one area where the current patent law as it's currently structured worked extremely well is the chemical pharmaceutical industry. And I think a lot of—their analysis of the patent system would find reasons for harmonization across international lines as a priority, improving PTO funding as a

priority. But reform of standards, remedies, et cetera, they think all that part—all those parts of the law are just fine. So there may well be other segments, too, of the economy where the patent law is working well.

MR. : One place where, if it weren't for the strength of the plaintiffs' bar, it would really pay to put some effort would be in litigation reform. That would help in many fields, not just in—you could find a coalition of interests which would have to support that, but since so many members of the Congress are lawyers, something like half of them, and because there is so much money generated in the plaintiffs' bar generally, I sort of despair of success there. But it's something I would like to see.

MR. : So let me ask you a little more generally about the idea perhaps of redefining what is a technical requirement. As mentioned earlier, there is basically no technical requirement in the patent law now, and, for example, the latest innovation is storyline patents. So one application which is now in process at the USPTO is for a process of relaying a story having a unique plot so that the person applying for this, the father of this is Andrew F. Knight, who published an article in the Journal of the Patent and Trademark Office Society, stating that basically any arguments about software patents can be applied to a storyline because there's no technical—technical advancement requirement or physical innovation requirement.

So there seems to be some clamoring for such a thing. Is it worth reconsidering what—or defining more clearly what a patent could cover, not in terms of software but in terms of sort of technical innovation?

MR. : Well, I implied, in fact, I think I said that I think it should involve science and technology, or maybe not even science, [inaudible] abstract. But patents should be about technology. Now, that leads itself to the kind of dodge that we saw in the creation of software patents. You use a general purpose computer, put your program in, and then it's a special machine. That was somewhat unfortunate, probably, although you could see in some of the earlier cases, like the oscillator case [inaudible] made a little sense there.

I don't—I really think we're getting into a treacherous area when we say all human thought should be subject to patenting if it's useful and non-obvious and so forth.

MR. : So how can we draw the line?

MR. : Well, I think you've got to leave it to the courts to draw the line. I think they should. I think they will.

MR. : Probably have, but [inaudible] the line.

MR. : After all, to a certain extent it's in the Constitution.

MR. CONNER: Hi, my name is Chris Conner (ph). I'm a student at the University of Maryland, and I'm just wondering what your thoughts are on the sort of patent ecosystem in the free and open-source software community and how the legal structure is changed when we're not talking about domestic companies shipping software solely within the U.S. and instead communities of people, possibly including companies here and abroad, contributing to a software project with not a single owner and then possibly having servers distributing it for free that are overseas.

MR. : Do you want to do this? Do you want me to do it? Okay, I'll do it.

Open-source developing models are a reality in the software industry. They're growing. The way the open-source developing models evolved at their inception was largely taking cognizance of copyright law and really not very much cognizance of patent law.

As more and more patents have popped up in the software area, that's a harder thing to ignore, so one of the things the open-source community is now doing is trying to work to develop—to redefine its license [inaudible] going on, to try to figure out how to incorporate [inaudible] considerations.

I think the abundance of patents, software patents out there does [inaudible] risk, right? So a lot of our lives are about risk avoidance. It does create risk in how you use open-source products because it's very hard to know who has incorporated what element and whether or not it's patented. So it's a level of uncertainty.

But the flip side of that is that open source is an existing and will-stay-around element of this stuff, where industry—though I think we as an industry as a whole are going to have to figure out [inaudible].

MR. : The first thing you have to recognize, which I'm sure you probably do, is that the free and open-source community is a very diverse community. It's almost like Christianity. There's all different people of different sizes, shapes, and desires, and in my list of people who [inaudible]; it includes large software companies, it includes small software companies, it includes individuals. So there are various actors in the free-software world who have various opinions about how patents [inaudible].

My premise, when I talk to people about the impact of having free software, is that patents have a harmful effect on free software, but a less harmful effect on free software than they have on non-free software, and that's because one of the strongest weapons a patent owner has is their power of an injunction, and the likelihood of getting an injunction against a closed-source software distributor who's someone easy to find, easy to get in court, easy to shut down, is much more easy to get than a distributed software [inaudible] that can come from 50,000 different serves across [inaudible]. It's easier to shut down Napster than it is LimeWire, okay? And LimeWire is the free software model and Napster is the non-free software model.

So having said that, you know—and the issues of risk, well, there's [inaudible] 600 new software patents coming out every week, so there's no one in the world who can guarantee you that their software that they've given you today won't infringe some patent next Tuesday. And so the issue then is how to transfer risk to a least-cost risk handler, usually through software code it's because the proprietary distributor will assume that risk as part of the price that they bake into the license. Free-software companies have this as a stand-aside product or an additional product you can buy, which if you want some transfer of risk to them for IP infringement, patent infringement, what have you, you can get that in the marketplace.

So my basic premise is that patents do have a negative effect on free and open-source software, but relatively less negative effect than they do on non-free software.

MR. : I hate to disagree with Dan [inaudible] I think he knows a lot more about it than I do. But it seems to me what we see in the marketplace—the

problem with open-source software is that it is an uncapitalized distributed operation, so it's basically in itself defenseless against assertions of [inaudible]. But what's complicated is there are many businesses, many big companies that are making a living off complementary products and services, like IBM.

Now, IBM has been criticized for not offering to indemnify its customers. In fact, IBM won't even distribute [inaudible] can't do it and [inaudible] the liability that goes with that. So there's this [inaudible] highly critical of IBM for not taking [inaudible] and comparing it to Microsoft assuming the risk of patent infringement [inaudible]. So we're getting a war on indemnification [inaudible] a very poorly developed insurance market for [inaudible]. A lot of companies have tried to get in it and gotten burned. There are big moral hazards, adverse selection problems. But the fact that, you know, some companies are trying to do it and there's some studies, mostly in Europe, of what's going on in the industry helps shed light on this [inaudible].

MR. : There's a question there.

MS. : My question is for you, Mr. Dam. Margot Korbay (ph). How much, if at all, do you think that the $200 million, 12-year antitrust lawsuit filed against—

MR. DAM: I can't quite hear you.

MS. : The two-year—12-year, $200 million antitrust suit filed against IBM by the Federal Government impacted IBM's patent strategy?

MR. DAM: I don't know that it did. You're talking about the one that was settled around 19...

MR. : '81.

MR. DAM: '81, right.

MS. : Yeah.

MR. DAM: When Baxter was at the Antitrust Division. I think it had a big impact on—it made IBM management timid about many things.

MS. : Oh, yeah.

MR. DAM: But I'm not—I wasn't there then. I didn't get to IBM until '85. And I am not—I never thought of any connection between that and its patent policy. The cross-licensing goes way, way back to the time when there were, you know, the seven sisters, or whatever they called them—I guess there were eight of them—and the time when the Japanese began to get into the business. So it probably goes back to the middle sixties.

So, clearly, the antitrust case didn't have any impact on the initial cross-licensing within computer hardware companies, but I don't know—do you have some suggestion as to how it would have impacted? I've just never thought about it.

MS. : I arrived about right in the heat of the antitrust lawsuit, and I left about five years after you did. What we saw in software development was much more emphasis on the IBM suggestion program and the patent program as employees, trying to keep everything internal, and much more reliance on trade secrets.

MR. DAM: Much more reliance on trade secrets?

MS. : Right. The fact that the State Street Bank case had not been even heard yet really didn't have any impact. The way we got around everything was with a trade secret.

MR. : Well, IBM filed an industry [inaudible] in 1981. So I've often wondered [inaudible]—

MS. : Yeah.

MR. : Was there any change in IBM's [inaudible] policy [inaudible]?

MS. : The other thing that's of interest is that years ago, as a result of the war, the Federal Government decided to make both IBM and AT&T essentially open up their entire patent portfolio to Japan for free. So imagine what that did in terms of the playing field of technology for the United States versus Japan. That was a major impact.

MR. : How long was that in effect?

MS. : Right after the war.

MR. : Right after World War II?

MS. : Yeah, yeah.

I have a second question for Judge Rader's clerk, former clerk. How much do you think the existence of the extremely effective union at the PTO has impacted the pendency rate and everything else negative going on at the PTO regarding patents?

MR. RAVICHER: Let me just clarify. I didn't clerk for Judge Rader. He was my professor, and I served him for a summer, but I didn't formally clerk for him in that capacity. But the Patent Examiner Corps Union—that's the one you mean, not the union of other Patent Office employees?

MS. : Are you a former examiner?

MR. RAVICHER: No.

MS. : Okay. I'm a former examiner. Very, very strong union there that's against any kind of modernization, automization, and obviously anything that would increase the amount of time spent for—spending time on a patent. Obviously they want more time, lower productivity rates, why we've got a two- and three-year pendency.

MR. RAVICHER: And so your question is? I'm sorry.

MS. : So how would you think that that has impacted things? Why doesn't anyone look at what they can do in terms of the union? Obviously, the union does not want to have anything that would automate or increase productivity there.

MR. RAVICHER: Well, I think the examiners get 99 percent of the blame and deserve 1 percent of it. I have never been an examiner, but I have known several of them, and I think that they do their best job possible, but they're subject to the pressure that they have, the economic pressures we talked about earlier, about the quota system and the review pressures. And I think that has a very harmful effect because at the end of the day, the decision the examiner makes should be based 100 percent on technology. Is this new? And is this un-obvious? Not on any other factor as far as, well, I need to get this done by the end of the month in order to meet my quotas, or what have you. And anything that takes the examiner away from being able to make a purely 100 percent technological decision I think introduces error into the system, and error is waste.

MR. KLEMENS: Ken, you had a—

MR. DAM: Well, I'm not—this is not responsive, but one thing that astonishes me here is not a single person has talked about what is very prominent in the academic literature and very much of concern in many industries, and that is, profusion of patents creates a problem of getting together what you need to create new products. Maybe that's because the software industry is still so dispersed and there's a lot of ideology in the industry and so forth.

But in many industries, this is alleged to be a real problem, but there are solutions. Cross-licensing is one. Patent pools are another. There's something called patent clearinghouses. And this comes up in this Proteomics report that I recommended to everyone who's interested at all in biotech. But it's also interesting that tomorrow and Friday there is a two-day OECD conference here in Washington—it's not open to the public. It's a working conference to look at how the biotechnology industry across the developed world can deal with this problem of the rapid growth in the number of patents in that industry. And there are examples, and the examples come from the software industry. There's the JPEG LA approach, if some of you may be familiar with that.

MR. : MPEG.

MR. DAM: Yeah, MPEG, excuse me. MPEG LA—

MR. : JPEG is the one that's having the [inaudible].

MR. DAM: Excuse me. Right. MPEG LA approach that they're going to—they've got a representative who's going to describe it. I know I have to chair that session. There is apparently a DVD alliance or consortium of some kind. And there's something in the G-3 area.

So there are mechanisms out there which would be one way of mitigating some of the problems that we're discussing.

MR. KLEMENS: Okay. Well, we're out of time, so if anyone would like to make a final comment or—

MR. : Thank you.

MR. KLEMENS: That sounds like a good one. Thank you.

[Applause.]

- - -