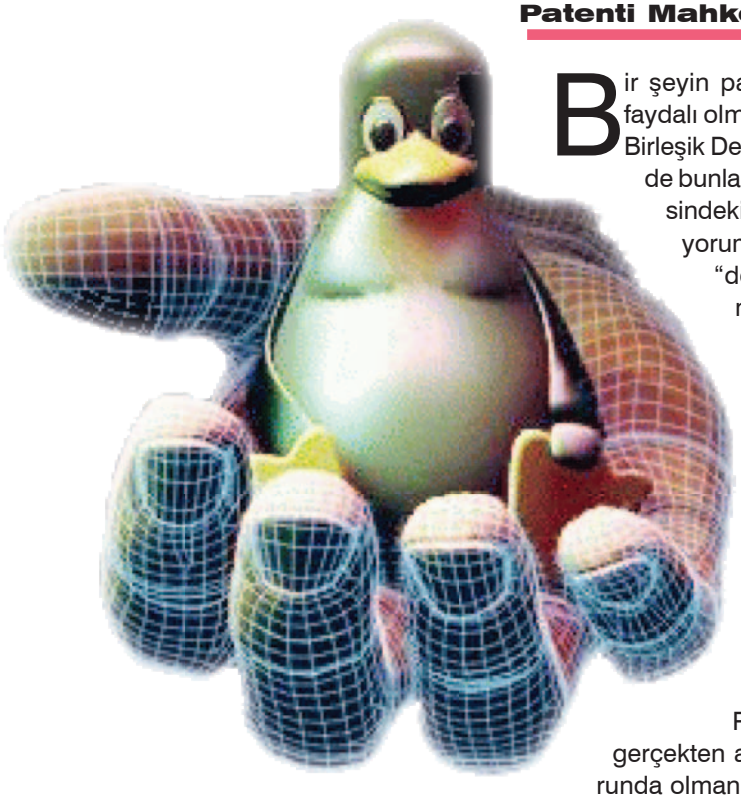
YAZILIM PATENTLERİ TEHLİKESİ[1]

2. Bölüm*

Richard M. Stailman



Patenti Mahkeme Aracılığıyla Geçersiz Kılmak

Bir şeyin patentlenebilmesi için, sözümona o şeyin yeni ve faydalı olması, bunun yanında açık olmaması gerekiyor. (Yani Birleşik Devletler'de böyle söylüyorlar; sanırım diğer ülkelerde de bunlara benzer sözcükler kullanıyorlardır.) Tabi patent ofisindekiler bu "yeni" ve "açık olmayan" laflarını farklı şekilde yorumlarlar. Onlara göre "yeni" şu anlama gelmektedir: "dosyalarımız arasında bulunmayan"; "açık olmayan"ın anlamıysa şudur: "I.Q. seviyesi 50 civarlarında olan biri tarafından pek kolay anlaşılamayan".

Birleşik Devletler'de verilmiş yazılım patentleri üzerine çalışmalar yapan biri -ya da en azından, bir zamanlar yapardı; hala bunları takip edebiliyor mu bilmiyorum- bu patentlerin %90 kadarının "Crystal City testi"ni geçememiş olması gerektiğini söylemişti. Yani, patent ofisindekiler, her gazete büfesinde bulunabilen şu bilgisayar dergilerinden herhangi birine şöyle bir gözatsalar, bu fikirlerin çoktan bilinen şeyler olduklarını kolaylıkla görebilirlerdi.

Patent ofisi o kadar aptalca işler yapar ki, bunların gerçekten aptalca olduğunu anlamanız için işi öğrenmek zorunda olmanıza bile gerek kalmazdı. Üstelik bu sadece yazılım patentleriyle sınırlı bir şey de değil. Bir keresinde, ünlü Harvard fare patentini görmüştüm. Bu patent Harvard'a, kansere neden olan bir gen kullanılarak, bir farenin genetik olarak incelenmesi üzerine verimişti. İncelemede kullanılan genin kansere yol açtığı zaten biliniyordu. Dahası, bu iş için seçilen farenin cinsi ve uygulanan teknikler de önceden biliniyordu. Ama aldıkları patent, kansere yol açan herhangi bir genin, herhangi bir yöntem kullanılarak, herhangi bir memelinin vücuduna yerleştirilmesini

*Bu yazının 1. bölümü Elektrik Mühendisliği Dergisi'nin 425. sayısında yayınlanmıştır.

1. 25 Mart 2002'de Stallman'ın Cambridge Üniversitesi'nde verdiği konuşma. Emre Uğur, İnan Kanbur ve İzlem Gözükeleş tarafından Türkçe'ye çevrilmiştir.

kapsıyordu. Bu durumun ne kadar gülünç olduğunu görebilmek için genetik mühendisi olmanıza gerek yok tabi. Bu "aşırı hak isteminin"-nin gayet normal bir şey olduğunu duymuştum. Hatta bazen Birleşik Devletler patent ofisinin, patent için başvuranlara, taleplerini daha geniş tutmalarını tavsiye ettiği bile oluyormuş. Esasında, taleplerinizi, önceden yapılmış ve çok bilinen bir işle çakışmadığı sürece istediğiniz kadar genişletebilirsiniz. Akıl alanında ne kadar arazinin yanınıza kar kalabileceğini görebiliyor musunuz?

Programcılar bir çok yazılım patentine baktıklarında şöyle derler: "bu gülünç derecede açık bir şey!" Oysa patent bürokratları/kırtasiyecileri, programcıların ne düşündüklerini umursamamakla ne kadar haklı olduklarını kanıtlayacak her türlü bahaneye sahiptirler. Cevapları hazırdır: "Bütün bunları 10-20 yıl öncesinin şartlarına göre düşünmeniz gerekiyor." Sonra sonra farkettiler ki, bir şeyi ölümüne konuşurlarsa, er geç kafanız karışabilir. Birçok parçaya bölünüp yeterince incelenirse, her şey karmaşık görünebilir. Bir süre sonra, 'açık' ve 'karmaşık' kavramları birbirine geçer; neyin açık, neyin karmaşık olduğuna nasıl karar verildiğini karıştırırsınız. Ve bu aşamada patent bürokratları size, patent sahiplerinin hepsinin ne kadar harika insanlar olduklarını anlatırlar; onların, yaptığımız işleri yönlendirme yetkilerini sorgulayamayız.

Yargıçların bu 'açıklık-karmaşıklık' konusunda biraz daha ciddi olacakları muhtemeldir. Fakat buradaki sorun, mahkemeye başvurmanın maliyetinin milyon dolarları bulmasıdır.

Davalının Qualcomm olduğu bir patent davası biliyorum. Qualcomm kaybetmişti. Hatırladığım kadarıyla karar, 13 milyon Amerikan Doları ödenmesi olmuştu. Bu paranın büyük bir kısmı iki tarafın avukatlarına gitmişti, davacıya ancak bir kaç milyon dolar kalmıştı.

Geniş bir açıdan bakacak olursak, bir patentin geçerliliği tarihi rastlantılara bağlıdır. Tam olarak neyin ne tarihte yayımlandığı, geçmişte yayımlanmış şeylerin ne kadarının bulunabildiği, ne kadarının kayıp olduğu, ve bunun gibi bir çok durum bir patentin geçerli olup olmadığı sorusunun yanıtının verilmesine yardımcı olur.

Doğrusu, *British Telecom*'un "telefonla erişimle hiperlinkleri takip etme" patentinin 1975 yılında uygulamaya konulmuş olması oldukça garip bir durum. Çünkü *Info* paketini ilk defa olarak 1974 yılında ben geliştirmiştim. *Info* paketi hiperlinkler boyunca hareket etmenizi sağlar; ve o günlerde insanlar sisteme ulaşmak için telefonları kullanıyorlardı. Görülüyor ki, bu patent alınmadan önce, ben kendim benzer bir iş yapmışım. Bu benim hayatım boyunca ürettiğim ikinci patentlenebilir fikir.

Bununla birlikte, bu durumu ispatlayacak herhangi bir kanıtım olduğunu sanmıyorum. Bunun yayımlamaya değecek kadar ilginç bir şey olduğunu bile düşünmemiştim. Üstelik, hiperlinkleri takip etme fikrini, Englebart'ın editörünün demosundan almıştım. Yayımlanmaya değer bir fikri olan biri varsa, o da Englebart'tır. Yaptığım işe "zavallı adamın hypertexti" adını vermiştim; çünkü işi, TECO bağlamında yapmak durumunda kalmıştım.Yaptığım şey, Englebart'ın hypertexti kadar başarılı değildi, ama en azından dokuman tarama işi için iyi bir araçtı; zaten işin bütün amacı da buydu. Telefonla erişim konusuna gelince: evet ortada gerçekten telefonla erişim hatları vardı ve sisteme bunlar sayesinde ulaşılıyordu. Ama bunun, işin diğer kısmıyla (zavallı adamın hypertexti) hiçbir alakası yoktu. Sırf ortada telefonla erişim hatları var diye de makale yayımlamayacaktım herhalde.

Bu işi tam olarak hangi tarihte yaptığımı söyleyebilmem pek olanaklı değil sanırım. Herhangi bir şekilde, bu işin yayımlanmış olma ihtimali var mı peki? Aslında, ARPANET'e gelmeleri ve makinelerimize girmeleri için bir kaç kişiyi davet etmistik -böylelikle *Info*'yu kullanarak dokuman taraması yapabilecekler ve olan biten hakkında fikir sahibi olabileceklerdi. Eğer bir şekilde bize sormuş olsalardı, telefonla erişimimiz olduğunu anlamış olacaklardı. Gördüğünüz gibi, tarihsel rastlantılar bir işin önce kimin tarafından yapıldığını belirlemede oldukça etkili olabliyor.

Hypertext konusunda Englebart tarafından yapılmış bir yayım şu an mevcuttur. Ve muhtemelen, davalılar bu yayımı mahkemede kullanacaklardır. Bununla birlikte, bu yayımda, sisteme bağlı telefonla erişimli hatlardan bahsedildiğini sanmıyorum; o yüzden mahkemede yeterli olup olmayacağı kesin değil.

Patenti mahkeme aracılığıyla geçersiz kılma olasılığı bir seçenek. Fakat masrafının çok olmasından dolayı, çoğu zaman bu seçenek kullanılmamaktadır, elinizde patentin geçersiz olduğunu gösteren kesin kanıtlarınız olsa bile. Sonuç olarak, geçersiz bir patent, gerçekte asla varolmaması gereken bir patent (ki aslında patentlerin bir çoğu böyledir) tehlikeli bir silahtır. Geçersiz bir patentle üzerinize gelen birileri başınıza gerçekten büyük sorunlar açabilir. Benzer bir işin daha önce yapıldığını gösteren bir kanıtla bunları uzaklaştırmaya çalışabilirsiniz. Korkup kaçacakları garanti değildir ama. Şu şekilde de düşünebilirler: "Pekala, sadece blöf yapıyorsun, mahkemeye gidemeyeceğini biliyoruz; buna gücün yetmez. O yüzden, her şekilde, seni dava edeceğiz."

Şimdiye kadar bahsettiğim yöntemlerin üçü de zaman zaman kullanabileceğiniz seçeneklerdir, fakat çoğu zaman bunları kullanamazsınız. O yüzden sürekli olarak patentlerle karşı karşıya kalırsınız. Ne zaman bu üç seçenekten birini kullanılabilir bulsanız, hemen arkasından başka bir patent çıkar karşınıza, onun arkasından da başka bir tane... İş, bir mayın tarlasından geçmeye benzemeye başlar. Attığınız her adım, aldığınız her dizayn kararı, bir patente denk gelmeyecektir muhtemelen, böylelikle bir kaç adım atabilirsiniz ve büyük bir ihtimalle bir patlama olmayacaktır. Fakat, program büyüdükçe, hiçbir patente denk gelmeden, mayın tarlasını boylu boyunca geçme ve istediğiniz programı geliştirme şansınız gitgide azalır.

İnsanlar bana diğer alanlarda da patent sisteminin olduğunu söylerler, ve yazılım alanının bu sistem dışında tutulması gerekliliğinin nedenini sorgularlardı. Buradaki tuhaf varsayıma dikkat edin lütfen: hepimizin patent sisteminin sıkıntısını çekmesi gerekiyormuş bir şekilde. Bu tam olarak, şöyle bir şey söylemeye benziyor: "Bazı insanlar kansere yakalanıyor. Neden sen bir istisna olmalıymışsın ki?" Gördüğünüz gibi, kansere yakalanmamış her insan iyi bir şey demektir.

Fakat, bunun ardında, daha az önyargılı bir soru var, iyi bir soru: Yazılım, diğer alanlardan farklı mı? Farklı alanlarda, farklı patent politikaları mı uygulanmalı? Eğer öyleyse, neden?

Cevaplamama izin veriniz: patentler farklı alanlarla farklı şekillerde ilgilidir, çünkü patenteler farklı alanlardaki ürünlerle farklı şekilde ilgilidir.

İlaç sanayiini ele alalım örneğin: bir kimyasal formül için patent alabilirsiniz, ve patent sadece ve sadece bir ürünü kapsar. Yeni bir ilaç varolan bir patentle çakışmaz. Bu yeni ürün için birilerine bir patent verilecekse eğer bir şekilde, bu kişi ürünü geliştiren kişi olacaktır.

Bu, patent sisteminin özündeki fikre uyan bir durumdur: yeni bir ürün geliştiriyorsanız şayet, patentini de alacaksınız demektir. Her ürün başına sadece bir patent düşer ve bu patent ürünün dayandığı fikri kapsar; ana fikir budur. Bazı alanlar için, bu durum, gerçeğe oldukça yakındır; fakat diğer alanlarda gerçeklikten çok uzaktır.

Yazılım sanayii ikinci duruma bir örnektir: bir program, bir çok patentle çakışır. Bunun temel nedeni, yazılım paketlerinin genellikle gerçekten çok geniş ve kapsamlı olmasıdır. Bir sürü farklı fikir birarada kullanılır. Eğer program yeniyse ve gerçekten bir yerlerden kopyalanmamışsa, büyük ihtimalle bilinen fikirlerin farklı bir birleşimi kullanılarak üretilmiştir. Ve tabi ki bu fikirler yeniden koda dökülmüştür, çünkü sadece bu fikirlerin isimlerini söylemek ve büyülü bir şekilde çalışmalarını beklemek bir işe yaramayacaktır. Bütün fikirleri yeniden koda dökmeniz gerekir. Bütün fikirleri belli bir kombinasyona göre yeniden koda dökmeniz gerekir.

Sonuç olarak, yeni bir program yazarken bir çok fikirden yararlanırsınız. Bu fikirlerden herhangi biri başka bir şahıs tarafından patentlenmiş olabilir. Hatta bu kişi, kullanıdığınız fikirlerden birkaç tanesinin birden patentlerini elinde bulunduruyor da olabilir. Bir fikrin birden fazla, farklı ifade şekilleri olabilir ve bunların herbiri çeşitli insanlarca patentlenmiş olabilir. Gördüğünüz gibi, programınızda başkaları tarafından patentlenmiş binlerce nokta bulunması oldukça yüksek bir olasılık.

Bu durum gösteriyor ki, yazılım patentleri meselesi, yazılım geliştirme işinin önünü tıkama yönünde yol almaktadır. Eğer gerçekten "ürün başına sadece bir patent" kuralı bu alanda da geçerli olsaydı, o zaman bu yazılım patentleri, yeni ürünlerin geliştirilmesi konusunda bu denli engel teşkil etmezdi. Çünkü geliştirdiğiniz yeni ürün, başkası tarafından önceden patentlenmiş olmazdı. Fakat birçok farklı fikrin birleşiminden oluşan yeni bir ürünün, bir süre sonra, bir kısmının veya tamamının, başkaları tarafından önceden patentlenmiş olduğu ortaya çıkacaktır.

Bu, neden yazılım patentlerinin yazılım süreçlerini engellemeye meyilli oldugunu açıklar. Eger her ürüne bir patent ilkesi geçerli olsaydı, bu patentler ürünlerin gelişimini engellemeyecekti çünkü yeni bir ürün geliştirildiği zaman başka biri tarafından patentlenmemiş olacaktı. Fakat bir ürün değişik fikirleri içerdiği zaman, sizin yeni ürününüzün (bir kısmının ya da tümünün) başka birisi tarafından patentlenmiş olma ihtimali oldukça yüksektir.

Gerçekten bir patent sistemini artan yaratıcılığın olduğu bir alana empoze etmenin o alandaki gelişimi yavaşlatacağını gösteren ekonomik araştırmalar bulunmaktadır. Yazılım patentlerinin olması gerektiğini savunanlar şunu söyler: "Tamam, problemler olabilir, fakat tüm bunlardan daha önemlisi, patentler yaratıcılığı ödüllendirir ve bu, neye neden olursa olsun çok önemlidir." Çok aptalca olduğu için tabii ki bunu yüksek sesle söylemiyorlar, fakat örtülü olarak sizin anlamanızı istedikleri şudur: Patent sistemi gelişimi ödüllendirdiği sürece, tüm bedellerin üzerindedir. Fakat, aslında gelişimi ödüllendirdiğine inanmak için bir neden yok. Şu anda elimizde patentlerin gelişimi nasıl geciktirebildiğini gösteren bir model var. Bu modelin uygulandığı durum, yani artımlı yaratıcılık durumu, yazılım alanını oldukça iyi tarif ediyor.

Neden yazılım, spektrumun bu kadar aşırı uçlarında yer alıyor? Çünkü yazılım yaparken, biz idealize edilmiş matematiksel nesneleri geliştiririz. Karmaşık ve büyük bir kale yapabilirsiniz, ve onu ince bir ipin üzerine koyabilirsiniz. Orada duracaktır çünkü fiziksel bir ağırlığı yoktur. Diğer alanlarda çalışma yapılırken, maddenin yada fiziksel nesnelerin zorlukları ve inatçılıkları hesaba katılmak zorundadır. Madde yapması gerekeni yapar. Onu modellemeye çalışabilirsiniz, ama eğer gerçek davranış sizin modelinize uymuyorsa şansınıza küsün, çünkü asıl meydan okuma gerçekten çalışan fiziksel nesneler yaratabilmektedir.

Bir while deyiminin içine if deyimini yazarken, if deyiminin belli bir sıklıkta salınım yapıp yapmayacağı ve while deyimine sürtünerek onu parçalamayıp parçalamayacağı üzerine endişelenmem. Aynen belli bir sıklıkta salınıp yaparken değişkenin değerini tümevarım yoluyla değiştirip değiştirmeyeceğin endişe etmemem gibi. If deyiminin ne kadar akım çekeceği, sonrasında while deyiminin içerisindeki ısıyı yok edip etmeyeceği ya da while deyiminde bir voltaj düşüşünün if deyiminin çalışmasına etkisi üzerine endişelenmem. Bu programı tuzlu su ortamında çalıştırırsam, tuzlu suyun if deyimi ile while deyimi arasına girip aşınmaya yol açıp açmayacağı üzerine de endişelenmem. [Bu olasılıklar sıralanırken izleyicilerden kahkahalar yükselir.]

Herhangi bir değişkeni ısınıp ısınmadığını düşünmeden yirmi kere kullanabiliriz. Değişkenin ne kadar kapasitansını olduğuyla yada değerine ulaşması için ne kadar zaman şarj edileceğiyle de ilgilenmeyiz.

Programı yazarken her kopyayı fiziksel olarak nasıl birleştirebileceğim yada while deyiminin içerisine bir if deyimi yazmak için nasıl erişebileceğim de endişelenmem gereken bir problem değildir. Oraya koyduğum if deyimi bozulursa ya da kırılırsa, onu oradan çıkarıp yerine yeni bir if deyimini nasıl yerleştirebileceğim de beni endişelendirmez. Yazılım yaparken beni kaygılandırmayacak çok fazla problem vardır ki, bu da temel olarak bir programın yazılımını fiziksel bir nesnenin tasarımından daha kolay yapar.

Bu garip görünebilir çünkü yazılım tasarlamanın ne kadar zor bir iş olduğundan, bunun ne kadar büyük bir problem olduğundan bahseden çok fazla insan görmüşsünüzdür. Aslında aynı sorundan bahsetmiyoruz. Ben fiziksel sistemlerle aynı karmaşıklıktaki ve aynı sayıda

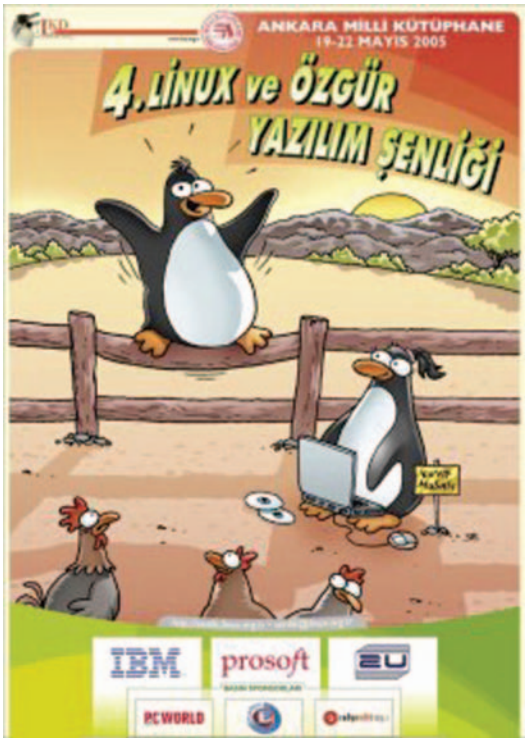
parçaya sahip yazılım sistemlerini karşılaştırıyorum. Bence yazılım sistemlerini tasarlamak fiziksel sistemleri tasarlamaktan çok daha kolaydır. Fakat bu farklı alanlarda çalışan insanlar aynı zeka seviyesinde olduklarına göre, daha kolay bir dalda çalışmak zorunda kalanlar ne olacak? Yetenekler sonuna kadar zorlanır. Aynı büyüklükteki iki sistemden biri basit, bu durumda ne yapacağız? Tabi ki basit bir sistemin büyüklüğünü on katına çıkaracağız, böylece zorlaşacaktır. Aslında şu anda yaptığımız da budur: fiziksel sistemlerden çok daha fazla parçaya sahip yazılım sistemleri üretmek.

İçerisinde milyonlarca farklı parça barındıran fiziksel bir sistemin tasarımı çok büyük bir projedir. İçerisinde bir milyon parçası olan bir yazılım projesi belki de 300,000 satırdan oluşur ki az sayıda insan iki yılda bu projeyi tamamlayabilir. Aslında bu çok da büyük bir proje değildir. GNU Emacs'ın tasarımı birkaç milyon parçadan oluşuyor. Bu projede yaklaşık bir milyon satır kod yazılmıştır. Bu proje çoğunlukla insanların boş zamanlarındaki üretimleriyle geliştirilmiş dolayısıyla herhangi bir parasal destek almamıştır.

Daha büyük başka bir tasarruf var. Fiziksel bir ürün tasarlanırken bir sonraki adım ürünün üretileceği fabrikanın tasarımını yapmaktır. Bu fabrikayı inşa etmek milyonlarca dolara malolurken bir programın kopyalayabilmek için "copy" komutu yeterlidir. Aynı kopyalama komutu tüm programları kopyalamak için kullanılabilir. CD'ye mi kopyalamak istiyorsunuz? Sorun yok, asıl CD'den çok sayıda CD yapabilirsiniz. Tüm CD kopyalama işlemleri için aynı ekipman kullanılır. Ürüne özel fabrikalar yapmanıza gerek yoktur. Bu, işlerin inanılmaz basitleşmesine ve tasarım maliyetinin muazzam azalmasına denk gelir.

Yeni model bir araba üretmek için 50 milyon dolarlık bir fabrika harcaması gereken bir otomobil firması için, avukatlar kiralayarak patent lisans görüşmeleri yapmak sorun değildir. Hatta açılan davalarla da başa çıkabilirler. Aynı karmaşıklıkta bir programın tasarımı 50 bin yada 100 bin dolara malolacaktır. Karşılaştırma yapılacak olursa, patent sistemiyle böyle bir durumda karşılaşmak yıkıcı olacaktır. Bir arabanın mekanik tasarım karmaşıklığına denk bir program yazmak bir ay kadar bir zaman alacaktır. Herhangi bir bilgisayar sistemi gerektirmeyen bir araba kaç parçadan oluşabilir... Bu, iyi bir tane asarlamanın kolay bir iş olduğundan söylemiyor[2]; sadece içinde çok da fazla parça içermediğini söylemek istiyoruz.

Bunu açıklamanın belki de en iyi yolu senfonilerle arasında benzerlik kurmaktır. Bir senfoni de uzundur, bir çok nota içerir ve müziğe dair pek çok fikir içerir. 1700'lerin Avrupa'sında hükümetlerin senfonik müzikteki ilerlemeleri ödüllendirmeye karar verdiklerini hayal edin. Kelimelere dökülebilen müzik ile ilgili tüm fikirleri patentleme hakkına sahip Avrupa Müzik Patent Ofisi adında bir kuruluşun kurulduğundan.



1800'lere gelin ve Beethoven olduğunuzu düşünün. Bir senfoni yazmak istiyorsunuz. Herhangi bir patenti çiğnemeyen kendi senfonizi yazmak "iyi bir senfoni" yazmaktan zor olacaktır.

Bu durum hakkında yakınacak olursanız, patentleri ellerinde tutan insanlar size şöyle diyeceklerdir: "Ah Beethoven, şikayet ediyorsun çünkü hiç orijinal fikrin yok. Tek yapmak istediğin bizim buluşlarımızı yürütmek." Beethoven'in birçok yeni fikri vardı, fakat algılanabilir bir müzik üretmek için varolan fikirleri kullanmak zorunda kalmıştır. Bir müziği dinleyenlere sevdirebilmek için, onların dinledikleri şeyi müzik olarak algılaması gerekir. Hiç kimse müziği tamamen farklı bir şekliyle yeniden icat edip, insanlara bunu sevdirecek kadar dahi değildir. Pierre Boulez, bunu yapmaya çalışacağını söylemiştir, peki şu anda Pierre Boulez'i kim dinliyor?

Kimse tüm bilgisayar bilimini, yepyeni ve bambaşka bir şekilde tekrar icat edecek kadar zeki değil. Eğer bunu yapa-

2. Bir arabanın iletim (**tranmission**) sisteminde yaklaşık olarak 300-400 birbirine benzemeyen parça bulunur ve iletim sistemi en arabaların en karmaşık bileşenlerinden biridir. Bir **transmission** tasarımı 6 aydan bir yıla kadar bir zaman diliminde tamamlanabilir, yapılması ve çalışır hale gelmesi ise büyük ihtimalle daha fazla zaman isteyecektir. Buna karşılık, 500-800 fonksiyonel parçası olan bir program 200-300 satır koddan oluşur ki bu da iyi bir programcı içim yazım, test ve yanlış ayıklamalar dahil bir gün bir hafta arasında tamamlanabilir.

bilen biri çıksa bile, bu kullanıcılara o kadar değişik gelecek ki, kimse bunu kullanmak istemeyecektir. Bir kelime işleyicisini inceleyecek olursanız, içerisinde yüzlerce farklı özellik bulursunuz.Yeni yaratıcı bir kelime işleyicisi geliştirecek olursanız, bu içerisinde yeni fikirlerin olduğu anlamına gelir; fakat aynı zamanda yüzlerce eski fikiri de barındırmak zorundadır. Eğer bunları kullanmak için izin verilmezse, yenilikçi bir kelime işleyicisi yaratamazsınız. Çünkü yazılım geliştirme işi o kadar büyüktür ki, sonuçta yeni fikirleri oluşturmak için yapay planlara gerek yoktur. Basitçe yazılım yapan insanlar vardır ve bunların kafalarında bazı yeni fikirler olacaktır. Eğer bir program yazmak istiyorsanız ve programın iyi olmasını istiyorsanız, bazı yeni fikirler bulursunuz ve bunları kullanmak için çalışırsınız.

Yazılım patentleri çıkmadan önce yazılımın içerisinde yeralan biri olarak biliyorum, daha önceleri, birçok yazılımcı, önemli olduğunu düşündükleri yeni fikirlerini, saygı göreceklerini düşündükleri için yayımlarlardı. Eğer fikirler çok ufak çaplı ise yada etkileyici değillerse yayınlamazladı, çünkü bu ters sonuçlar doğurabilirdi. Şu anda patent sistemi fikirleri açıklamamayı özendirir hale gelmiştir. Doğru, kodlarını saklarlardı ama eski günlerde kimse fikirlerini saklamazdı. Sonuç olarak kod, yapılan işin tamamına denk gelir. Kodu saklarlardı ve fikirleri yayınlarlardı, böylece çalışanlar hem kendilerini iyi hisseder hem de tanınırlardı.

Yazılım patentleri çıktıktan sonra kodlarını saklamaya devam ettiler, ve fikirlerini patentlediler. Gerçekten de fikirleri açıklama herhangi bir yönden teşvik edilmiyor. Bugün, eskiden gizlenen şeyler halen gizli tutuluyor, fakat eskiden yayımlanan ve bizim kullanabildiğimiz fikirler bugün patentleniyor ve 20 yıl boyunca ulaşılmazlıklarını koruyorlar.

Bir ülke bunu değiştirmek için ne yapabilir? Bu problemi çözmek için politikamızı ne şekilde değiştirmeliyiz?

Atağa geçebileceğimiz iki yer var. Birisi patentlerin dağıtıldığı yerler, yani patent ofisleri. Diğer yer ise patentlerin uygulama alanları, yapılacak şey ise patentlerin kapsamlarının sorgulanması.

Bir yol patentleri dağıtırken iyi kriterler koymak. Bu yöntem sadece, Avrupa'nın büyük kısmında olduğu gibi, daha önce yazılım patentleriyle ilgili bir kısıtlama olmayan ülkelerde uygulanabilir.Avrupa Paten Ofisi kuralları yazılım patentlememenin Avrupa için iyi bir çözüm olduğunu söylüyor, bizim yapabileceğimiz bunu pekiştirmeye çalışmak olabilir. Avrupa şu anda yazılım patentleri konusunda kendine bir yön çizmeye çalışıyor (Yönelimin daha geniş bir çerçevesi olduğunu ve bunların yazılım patentleri açısından önemli sonuçları olacağını düşünüyorum.). Yazılım fikirlerinin patentlenemeyeceğine karar vermek, Avrupa'nın büyük bir kısmını bu problemden kurtarır. Fakat ne yazık ki, Birleşik Krallık dahil bazı Avrupa Ülkeleri bu sorunu sadece kendi sorunları olarak görmektedir.

Bu yaklaşım Birleşik Devletler'de geçerli değildir. Çünkü Birleşik Devletler zaten çok fazla sayıda yazılım patenti elinde bulundurmaktadır ve patentlerle ilgili kriterlerdeki herhangi bir değişim halihazırda varolanları bir kenara itemez[3]. Bu nedenle Birleşik Devletler'de çözüme patentlerin uygulanabilirliklerini ve kapsamlarını değiştirerek ulaşabiliriz.Söyle söylemeliyiz: Donanımsal olarak patent yasalarını çiğnemeyen bir genel amaçlı bilgisayarda çalışan saf bir yazılım gerçekleştirim'i patent kapsamına alınamaz ve dava konusu olamaz. Bu da başka bir çözüm.

İlk çözüm, yani geçerli olan patent tiplerini belirlemenin baskın olduğu çözüm, Avrupa için iyi bir çözüm olabilir.

Birleşik Devletler'de ilk yazılım patentleri verilmeye başlandığı zaman, politik bir tartışma olmadı. Gerçekte, kimse bunu farketmedi. Yazılım alanındaki insanların çoğunluğu farketmedi bile. 1981'de Anayasa Mahkemesi'nin "curing rubber" işleminin patentlenebileceğine dair bir kararı oldu. Kural şuna dayanıyordu: "curing rubber" işlemi sırasında kullanılan aygıtların bir bilgisayar ve program içermesi, bu sürecin patentlenemez olduğunu göstermez. Bir sonraki yıl, patentlerle ilgili tüm patent başvurularında argümanlar değişti: Eğer bir süreç bilgisayar ve program içeriyorsa, o patentlenebilir olmalıydı. Sonuç olarak bilgisayar ve program içeren herşey patentlenebiliyor. Bu Birleşik Devletler'de neden iş yöntem patentlerinin ortaya çıktığını açıklıyor: çünkü iş yöntemleri bilgisayar üzerinde yürüyor ve bu onları patentlenebilir kılıyor.

Bu kural böylece gerçekleşti, ve sanırım doğal sıralamanın yeniden hesaplanması patenti de bunlardan ilki oldu.

80'li yıllar boyunca bunların farkında değildik. İlk defa 90'ların başında Birleşik Devletler'deki programcılar yazılım patentleri tehlikesi ile karşı karşıya olduklarının farkına varmaya başladılar. Ben bu alanda tüm

3. "Yazılım patentleri"nden bahsederken gerçekten neyi kastediyorum? Birleşik Devletler Patent Ofisi patentlerde yazılım patentleri ve diğer patentler gibi bir ayrım yapmamaktadır. Böylece, gerçekten de bir patent, eğer yazılım alanında uygulanabilinirse, size dava açılmasını sağlayabilir. O halde yazılım patentleri, yazılıma uygulanma potansiyeli olan ve potansiyel olarak bir program yazdığınızda size dava açılmasına yolaçan patentlerdir.

bunlar olmadan önceki çalışmaları da gördüm, bunlar olduktan sonrakileri de. 1990'dan sonra yazılım alanındaki ilerlemelerde herhangi bir hızlanmaya şahit olmadım.

Birleşik Devletler'de herhangi bir politik tartışma olmadı, fakat Avrupa'da büyük politik tartışmalar oldu. Birkaç yıl önce Avrupa Patent Ofisini kuran Munich Antlaşması üzerinde değişiklik yapmak istediler; antlaşma yazılımın patentlenemeyeceğine dair bir madde içerdiği için antlaşmayı yazılım patentlerini kabul edecek şekilde değiştirmek istediler. Özgür yazılım geliştiricilerin ve özgür yazılım kullanıcılarının başını çektiği topluluk durumun farkına vardı. Yazılım patentleri sadece bizi tehdit etmiyordu, tüm yazılım geliştiricilerini ve hatta kullanıcıları tehdit ediyordu.

Örnek olarak Paul Heckel'i verebiliriz; tehditleri Apple firmasını korkutmadı. Fakat Apple'ın müşterilerinin dava etmekle tehdit etmeye başladığında Apple durumu korkutucu buldu. Sonuç olarak müşterileri kazanacak olsa bile, tüm bu davalarla başa çıkamayacaklarının farkına vardı. Yazılım kullanıcıları da, geliştiricilere saldırmak için yada para sızdırmak için dava edilebilirdi. Tüm geliştiriciler ve kullanıcılar kolayca zarar görebilirdi.

Yasayı değiştirmek için yapılan baskıya karşı örgütlenmede başı çekenler Avrupa'daki özgür yazılım topluluğu oldu. Gerçekten de Avrupa ülkeleri patent düzenlemelerini ikinci kere reddetti. Daha sonra duruma Avrupa Topluluğu el koydu, topluluğun kuruluşları bu konuda ikiye bölündü. Bu kurumlardan biri, işi yazılımları ödüllendirmek olanı yazılım patentlerine karşı çıkıyordu; fakat bu konudayetkili değildiler. Yetkili Open Market Directorate (Açık Pazar Kurumu) idi ve kurumun başkanı yazılımın patentlenmesini savunuyordu. Halkın fikrine aldırmadılar ve yazılım patentlerine izin veren bir talimat önerdiler.

Fransız hükümeti yazılımın patentlenmesine karşı olduğunu açıklamıştı bile. İnsanlar Avrupa'daki diğer hükümetlerinin yazılımın patentlenmesine karşı çıkması için çalışıyorlardı, özellikle bunu burada, İngiltere'de başarmak çok hayatiydi. Avrupa'daki mücadelenin liderlerinden biri olan Hartmut Pilch'e göre ana darbe Birleşik Krallık patent ofisinden geldi. Birleşik Krallık patent ofisi yazılımların patentlemesinden yana bir eğilim sergilemişti. Yaptıkları bir kamuoyu yoklamasında, insanların çoğu yazılım patentlerine karşı çıkmıştı. Buna rağmen insanların onlarla aynı fikirde göründüklerini söyleyen bir rapor hazırladılar. Özgür yazılım topluluğu sonuçların halka ve kendilerine gönderilmesini istedi. Böylece sonuçları yayınlamak zorunda kaldılar fakat genel olarak insanlar sonuçlara karşı çıktı. Rapora baktığınızda, bunun Birleşik Krallık patent ofisi tarafından yayınlandığı hemen anlaşılıyordu.

Çok fazla saptırılabilecek bir şeye, "technical effect" (teknik etki) terimine başvurdular. Şöyle düşünmeliydik: Bir program ancak spesifik bir fiziksel harekete (physical acts) denk gelirse patentlenebilirdi. Eğer bu şekilde yorumlanırsa, problem çözülmüş oluyordu. Fakat problem şu ki bu terimi istediğiniz gibi saptırabilirsiniz. Bir program çalıştırarak elde ettiğiniz bir sonucu fiziksel bir sonuçmuş gibi açıklayabilirsiniz. Fiziksel bir sonuç diğerinden nasıl farklı olabilir ki? Tamam, o zaman bu sonuç bir hesaplamanın (computation) sonucu. Birleşik Krallık patent ofisinin önerdiği şey problemi tamamen çözüyormuş gibi görünse de, herşeyin patentlenebilmesinin yolunu açıyordu.

Aynı bakanlıktaki insanlar telif hakları konusuyla ilgilenmişlerdi, fakat bu konunun yazılım patentleriyle aynı insanların ilgilenmesi dışında ortak bir noktası yok.(Belki de "fikri mülkiyet" terimini kullanarak tüm bu konuları aynı yığında topladılar) Avrupa Topluluğu'nun yeni telif hakları yasası, Birleşik Devletler'deki Digital Millennium Copyright Act (DMCA) kanunu kadar korkunç olsa da, yasayı nasıl uygulayacağına karar verme konusunda ülkelere esneklik tanınmıştır. Birleşik Krallık direktifin uygulanması için en sert yolları öneriyor. Yasayı olabildiğince düzgün bir şekilde uygulayarak zararını en aza indirebilirsiniz. Birleşik Krallık bu yasanın kötü etkilerini mümkün olduğunca arttırmaya çalışıyor. Dizginlenmesi gereken belirli bir grup varmış gibi görünüyor - Department of Trade and Industry (Ticaret ve Sanayi Departmanı)-. Bunların yaptıkları işler gözlenmeli ve güçlenmeleri engellenmelidir.

Yazılım patentleri, geliştiricilerin ve tüm bilgisayar kullanıcılarının bir tür bürokrasi içinde ellerini bağlamaktadır. Bilgisayar kullanan iş çevreleri yazılımın patentlenmesinin başlarına nasıl bir iş açacağını farkederlerse, harekete geçip bunu durdurabilirler. İş çevreleri bürokrasi içerisinde sıkışıp kalmaktan hoşlanmazlar. Bazen tabi ki, bürokrasi önemli bir amaca hizmet edebilir.Birleşik Krallık hükümetinin belli bazı işleri bürokrasiyle kısıtlamasını isteyebiliriz, tıpkı hayvan taşımacılığında olduğu gibi[4]. Fakat yapay tekeller yaratmak dışında başka bir amaca hizmet etmiyorsa, yazılımların patentlenmesine karşı çıkmalıyız. Yöneticilerin yazılım patentlerinin nelere yol açabileceğinin farkında olmalarını sağlayıp, bu mücadelede onların desteğini almalıyız.

Savaş henüz bitmedi, hala kazanabiliriz.

4. Bulaşıcı hastalıkların yayılmasını engellemek için.