GRAFİK SANATLAR ÜZERİNE
Yazılar:
GRAFİK TASARIMCILAR MESLEK KURULUŞU

Kötü tasarım bir virüs gibidir: tasarım kusurları ve gizli hatalar

Bir tasarımı iyi yapan nedir?

Michael Parent
uxdesign.cc
Çeviri: **Sıla Tülü**

Tasarımcılar genellikle ürünün veya arayüzün kullanışlılığından, ne kadar iyi performans gösterdiğinden, düzeninin sezgisinden ve hatta belki de bazı hedef veya amaçlara yönelik performansından (örneğin tıklamalar gibi!) bahseder. Ancak tasarımcıların yalnızca bu özelliklere odaklanmaması gerekir. Tasarlanan bir ürün veya sistemin başarısız olma eğilimi, tasarımcılar için çok önemli bir husustur.

Tasarım ve kullanıcı deneyimi tartışılırken bu genellikle düşünülmez. Bu çabalar, normal kullanım koşullarının ne olduğuna odaklanır ve ardından bu koşullar altında deneyim ve performansı optimize etmeye çalışır. Bu koşullara bazen Optimal Tasarım Alanı (*Optimal Design Domain* – ODD) adı verilir.

Hatalara odaklanıldığında genellikle ODD dışındaki koşulları ve davranışları göz önünde bulundururuz. Aslında bu, iyi tasarımı değerlendirmenin başlıca yollarından biridir. Bir sistem ürününün ne sıklıkta hata verdiğine ve hatanın sonucunun ne kadar ciddi olduğuna bakmak, tasarımın kalitesi hakkında çok şey söyler. ODD'nin dışına bakarak, rutin olmayan koşulları, yanlış kullanım davranışlarını ve tasarım risklerini değerlendirmek, daha güvenilir ve dayanıklı bir tasarım oluşturmaya yardımcı olabilir.

İsviçre Peyniri Modeli. İşler ters gittiğinde.

İşler ters gittiğinde nedensel bir açıklama ararız. Ve işler *genellikle* ters gitmediğinden, ters gittiklerinde nedensel açıklama genellikle kullanıcıların davranışlarına odaklanır. Daha zarif açıklamalar aynı zamanda sistem dinamikleri ve sistemin «durumları» üzerine de odaklanabilir, ancak odak noktası sezgiyi takip eder ve hatadan hemen önceki olaylar üzerinde kalır. Bunlar **aktif hatalar** olarak adlandırılır. Aktif hatalar tam olarak onlardan beklediğimiz şeylerdir; anında hataya neden olan sorunlar, davranışlar, ayarlar ve seçimler.

Ancak hatanın tek anlık nedeni bu değildir. Beklemede veya uyku halinde olan ama büyük hasara yol açma potansiyeline sahip başka faktörler de vardır. Bunlar **gizli hatalar** olarak adlandırılır. Aktif hatalardan farklı olarak bunlar, yöneticilerin, mühendislerin ve tasarımcıların kararlarından ve seçimlerinden oluşur. Gizli hatalar, tasarım tercihleri yapılıp sisteme uygulanırken, aylar, bazen yıllar öncesinde meydana gelir.

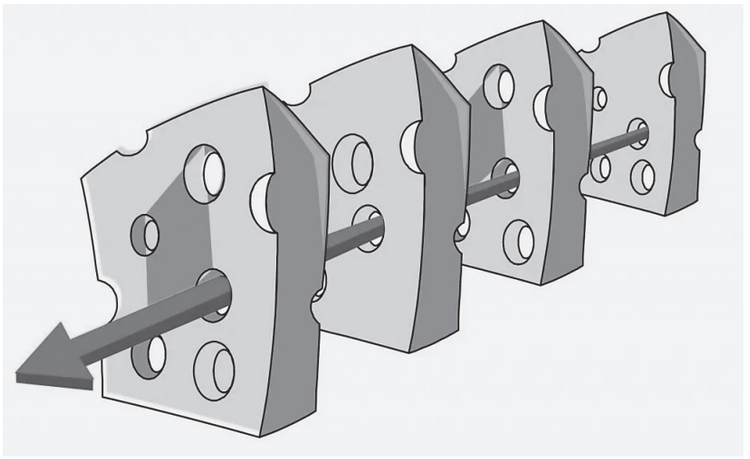
2000 yılında, İnsan Performansı uzmanı ve psikolog James Reason, sistem hatalarını anlamak ve gizli hataların etkisini göstermek için bir çerçeve sundu. Buna İsviçre Peyniri Modeli adını verdi. Model görece basittir. Herhangi bir sistem için, bir hatanın oluşmasını engelleyebilecek birçok katman vardır. Bu katmanlar yönetim kararlarını, tasarımı, makine performansını, kullanıcı davranışını, protokolleri ve güvenlik özelliklerini içerir.

Kısaca, İsviçre Peyniri Modeli, bir hataya katkıda bulunabilecek her faktörün uçtan uca bir örneğidir. Eğer herhangi bir katman doğru performans sergiliyorsa, hatanın yayılması engellenir. Ancak tüm katmanlara nüfuz edilmesi durumunda sistem hatası ortaya çıkar.

Bu modelin uygulanmasına örnek olarak HMS Titanic'i düşünün. Bu trajedi sadece bir buzdağını görememenin sonucu değildir, aynı zamanda geminin hızı, izlediği rota, yılın zamanı ve elbette geminin su geçirmez bölmelerindeki malzeme hatası gibi yönetim kararlarını da içerir. Bunlardan herhangi biri mevcut olmasaydı, bu trajedi muhtemelen yaşanmayacaktı.

İsviçre Peyniri Modeli tasarımcılar için önemli çıkarımlara sahiptir. Modelin mantığına göre iyi bir tasarım tüm hataların meydana gelmesini engelleyebilir. Bu pratikte %100 doğru değilse bile kesinlikle yakındır. İyi bir tasarım, potansiyel bir hatanın bir organizasyon içinde yayılmasını engelleyebilir.

Kaynak: Reason, 1997.



Yerleşik patojenler

Ancak bir tasarımcının her türlü hatayı, özellikle de ODD dışındaki koşullardan kaynaklananları önleyecek kadar sağlam bir tasarım yaratması pek olası değildir. Peki o zaman tasarımcılar, hataların ve gizli hataların önlenmesi için tasarım hakkında nasıl düşünmelidir? Bunun bir yolu, yerleşik patojen metaforu olarak adlandırılan bir şey aracılığıyla olabilir.

Yerleşik patojen, bir organizmada hastalıkların varlığını açıklayan tıbbi bir fikirdir. Basitçe ifade etmek gerekirse, tek bir patojen (bakteri, kanser hücresi, kanserojen, virüs) hastalığa neden olmaz, ancak organizmadaki patojenlerin sayısı arttıkça hastalığa yakalanma olasılığı da artar. James Reason bu metaforu gizli hatalar ve tasarım hataları hakkında düşünmenin yapıcı bir yolu olarak sunuyor. Patojeni tasarım kusurlarıyla, organizmayı ise bir organizasyonla değiştirin. Tasarım ne kadar iyi olursa, hatayla sonuçlanma olasılığı da o kadar az olur.

Reason şöyle açıklıyor:

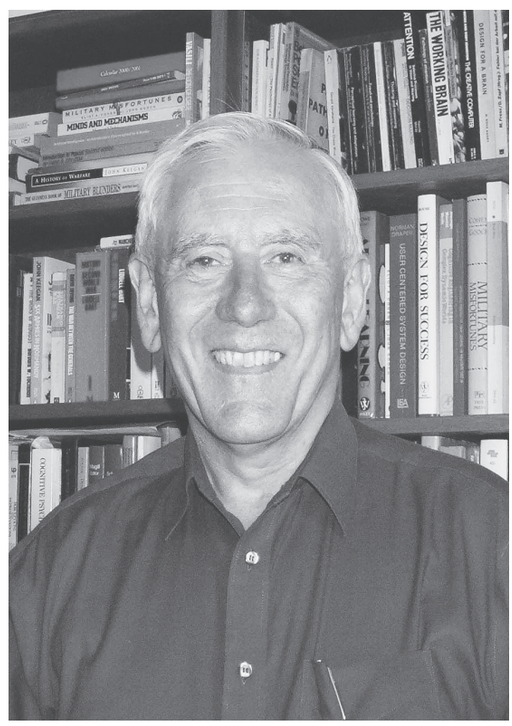
«Gizli hatalar, insan vücudu içindeki, dışsal faktörlerle (stres, toksik etkenler vb.) birleşerek hastalığa yol açan ‹yerleşik patojenlere› benzerdir. Kanserler ve kardiyovasküler bozukluklar gibi, karmaşık, savunulan sistemlerdeki kazalar tek bir nedenden kaynaklanmazlar. Felaket bir arızaya neden olmak için her biri gerekli olmakla birlikte tek başına yetersiz olan birkaç farklı faktörün öngörülemeyen (ve genellikle öngörülemez) birleşmesi yoluyla meydana gelirler.»

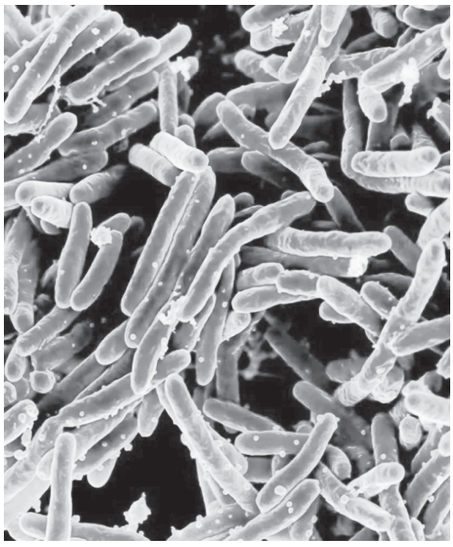
Bu metaforu yol gösterici bir ışık olarak kullanarak, gizli hataların nedenleri, etkisi, davranışı ve yayılması hakkında birkaç şey söyleyebiliriz.

İlk olarak, ve en belirgin olanı, bir sistem içindeki sorun olasılığı, patojenlerin (tasarım kusurları) sayısı arttıkça artar.

İkinci olarak, bir sistem ne kadar karmaşık, bağlantılı ve anlaşılmaz ise, patojen (tasarım kusurları) sayısı o kadar daha fazla olur. Bu durum birkaç nedenden dolayı doğrudur.

Kaynak: https://www.pressesdesmines.com/





Kaynak: News-Medical.Net

Kaynak: Airforcetimes.com/

Savunma yüklenicileri gibi gizli projelerde çalışan bir şirket düşünün. Tasarımcının kendisi en iyi tasarımı oluşturmak için ilgili tüm ayrıntılara sahip olmayabilir. Ayrıca, karmaşıklık kontrol edilemez, yalnızca yönetilebilir. Bir sistem aşırı karmaşıksa, sistem içinde bilinmeyen birçok etkileşim vardır. Etkileri de aynı şekilde bilinmemektedir. Son olarak, aşırı anlaşılmazlık ve karmaşıklık, iyi tasarım çıktıları değildir. Bir tasarımcı böyle bir çıktı oluştururursa, büyük ihtimalle birçok tasarım düşüncesi gözden kaçmış demektir.

Üçüncü olarak, daha etkili ve kârlı olan faaliyetler, aktif hataların sürekli olarak önlenmesi ve tespit edilmesi yerine, bu gizli hataların proaktif olarak belirlenmesini ve etkisiz hale getirilmesini hedefler. Kanseri önlemek, onu tedavi etmekten daha iyidir.

Dördüncü olarak, tüm olası hata tetikleyicilerini görmek ve ele almak imkânsızdır. Ancak patojenik riskler sistem bilgisi aracılığıyla değerlendirilebilir.

Zayıf çözümler

Tasarım kusurlarının ve yaklaşan hataların varlığı ışığında ne yapılmalı? Bu zorlukları ele almak için birçok seçenek mevcuttur. Ancak, bu seçeneklerden bazıları yüzeyde iyi fikirler gibi görünse de, aslında sisteme pek etkisi olmayabilir ve bazı durumlarda amaçlanan etkinin tam tersine neden olabilir.

İlk çözüm kullanıcı ve operatör eğitimidir. Bu çözüm muhtemelen tüm okuyuculara tanıdık gelir. Genellikle eğitim, kullanıcı başlatma sürecinin bir parçası olarak gerçekleştirilir. Yeni kullanıcılara ve operatörlere belirli bir sistemin nasıl çalıştığı öğretilir. Bu tür eğitimler eğitimli bir kolaylaştırıcı, bir akran veya konunun bir uzmanı tarafından gerçekleştirilebilir. Her birinin kendine göre artıları ve eksileri vardır, ancak burada daha önemli olan şey, eğitimin odak noktasının neredeyse her zaman aynı olmasıdır – sistemin *normal* kullanımı ve işleyişi. Kısacası eğitim, özellikle sistemin tanıtımıyla birleştirildiğinde, operatörü sistemi ODD dahilinde tutacak davranışları nasıl seçeceği konusunda eğitmeye odaklanır.

Gizli hatalar riskini azaltmaya yönelik bu yaklaşım açıkça hatalıdır. İsviçre Peyniri Modelinde daha önce belirttiğimiz gibi, gizli hataların etkisi büyük olasılıkla sistem ODD dışında çalışırken ortaya çıkar. Normal işleyişi mümkün olduğunca sağlamaya çalışmak önemli olsa da, bu, kaçınılmaz olarak ortaya çıkan gizli hataların riskini ele almada pek etkili değildir.

İlk bakışta değeri olan bir diğer teknik, tasarım hatalarını tespit etmek ve ele almaktır. Bunun, özellikle Arıza Modları ve Etkileri Analizi (*Failure Modes and Effects Analysis* – FMEA) aracının kullanılması gibi sistematik bir şekilde yapılması halinde, bir miktar etkisi olabilir. Bu araçların faydaları, işlerin nasıl ters gidebileceğini düşünmenize ve ardından etkileri ve nedenleri belirlemenize olanak tanıyarak sizi daha yaratıcı düşünmeye zorlamalarıdır. Yine de ODD paradigmasının dışında düşünmek hâlâ zordur. Tamamlanmış bir FMEA, bir sistem içindeki yerleşik patojenlerin sayısını önemli ölçüde azaltabilse de, hata durumunda dayanıklılığı, yanıtı ve iyileşmeyi geliştirme konusunda herhangi bir katkı sağlamaz.

Neden sadece bir sürü ek güvenlik önlemi eklenmiyor? Eğer gizli hata sorunları sistem anormal çalışırken ortaya çıkıyorsa, o zaman bu senaryolarda çalışmanın imkânsız olması için kısıtlamalar ekleyelim. Bu yaklaşım mantıklı görünse de aslında yukarıda belirtilen ilkelerin çoğunu ihlal etmektedir. Ne zaman karmaşık, anlaşılmaz ve işleyiş ile sonuç arasındaki ilişkiye aracılık eden sistemler yaratsak, aslında sisteme daha fazla patojen ekliyoruz. Bir örnek düşünelim. 737 MAX kötü tasarlanmış bir uçaktı. Uçağın geliştirilmesinde ve üretiminde zamandan ve paradan tasarruf etmek için mevcut gövde tasarımına daha büyük motorlar eklendi. Perdövites sorunlarını önlemek ve 737 MAX'i ODD koruması dahilinde tutmak için, burnun çok hızlı şekilde çok yükseğe çıkmasını önlemek amacıyla Manevra Özellikleri Arttırma Sistemi (*Maneuvering Characteristics Augmentation System* – MCAS) yazılımı kuruldu. Bu iyi bilinen bir şeydir. İyi bilinmeyen şey ise, uçağın seyir halindeki açısını değiştirmek veya MCAS'nin fitilinin çekilmesi dahil olmak üzere MCAS'yi devre dışı bırakan ek güvenlik önlemleridir. Bu önlemler, Boeing'i kötü bir tasarımla ilerlemeye cesaretlendiren şeylerdi. Aslında toplamda 346 kişinin ölümüne yol açan iki Boeing 737 Max kazasına (Lion Air Uçuşu 610 ve Etiyopya Havayolları Uçuşu 302) neden olan da MCAS korumasının kendisiydi. Kötü tasarım, güvenlik önlemleri oluşturarak aslında daha da kötüleştirildi.

Kaynak: Boeing.com/

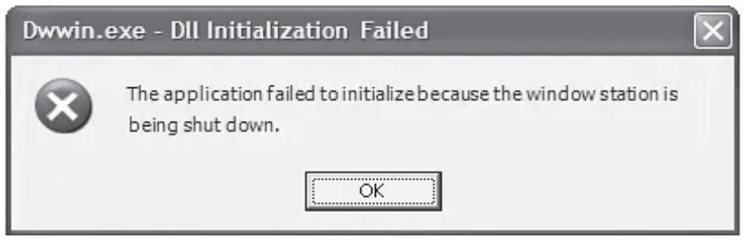
İyi çözümler

Gizli hataları ele alma konusundaki yaygın yanlış adımlara rağmen, bunların risklerini etkili bir şekilde ortadan kaldırmak için sunulabilecek çeşitli çözümler vardır.

Durum Temelli Eğitim. Kullanıcılara belirli bir çıktıyı elde etmek için sistemi nasıl sürdüreceklerini öğretmeye odaklanan geleneksel eğitimin aksine, durum temelli eğitimin çok farklı bir amacı vardır. Amacı, operatörlere alışılmadık senaryolar ve kriz durumlarıyla çalışma fırsatları sunmaktır. Havayolu pilotları bu disiplinin harika örnekleridir. Lisanslarını korumak için birkaç saatlik uçuş simülatörü eğitimi kaydetmeleri gerekir. Bu simülasyon eğitimi sadece rutin olmayıp, motor arızaları ve kabin basıncının düşmesi gibi zorlu senaryoların ele alınmasına da odaklanmaktadır. Bu durum temelli eğitimler, doğru davranışın seçilmesine yardımcı olabilecek kuralları ve düşünce kalıplarını pekiştirmeyi amaçlamaktadır. Daha önce UX Collective'de bilişsel süreçlerin farklı seviyeleri ve problem çözme hakkında çok daha fazla yazmıştım.

Simülasyon. Durum temelli eğitime benzer şekilde simülasyon, uç durumlara daha aşina olma fırsatı sağlar. Durum temelli eğitimde olduğu gibi simülasyonlar da belirli, nadir görülen durumların, bazen birkaç binlerce denemeyle kapsamlı bir şekilde değerlendirilmesine olanak sağlayacak şekilde oluşturulabilir. Simülasyon aynı zamanda olası gizli tasarım kusurlarını ortadan kaldırabilecek sistem dinamiklerinin araştırılmasını sağlaması açısından da değerlidir.

Simülasyon birçok biçimde gerçekleştirilebilir. Monte Carlo Simülasyonları ve Discrete Event Simülasyonları, modelleme sistemleri için en popüler olanlardan ikisidir. Bununla birlikte, oldukça karmaşık sistemler için, ajan tabanlı simülasyonlar, durum temelli eğitim ve simülasyonun



Pencere istasyonu kapatılmakta olduğundan uygulama başlatılamadı. Ve Kaynak: Freshsparks.com/

birleşimi yoluyla daha da büyük içgörüler ortaya çıkarabilir. Bir kriz egzersizi simülasyon platformu olan Conducttr buna mükemmel bir örnektir. Hükümetler ve özel kuruluşlar, farklı düzeylerde kontrollere, hedeflere ve sınırlara sahip geniş bir aktör ağı kullanarak kriz senaryolarını simüle edebilir ve potansiyel zayıf noktaları ortaya çıkarabilir.

Hata yönetimi. Gizli hataları yönetmenin bir başka uzman yolu da onlar için plan yapmaktır. Yazılım tasarımı bu tekniğin önemli bir kullanıcısıdır. Bu kadar çok satır, bu kadar karmaşıklık ve koda dayanan bu kadar çok sistem varken elbette bir şeyler ters gidecektir.

Programlamada hata yönetimi, kod yürütülürken ortaya çıkabilecek beklenmedik sorunlarla ilişkili kaçınılmaz zorlukları ele alan, programlamanın temel bir yönüdür.

Yazılım mühendislerinin en iyi uygulaması, hataları önceden tahmin etmek ve hatalar meydana geldiğinde bunları yönetmek için kurallar oluşturmaktır.

Uyarlanabilir Durumsal Modlar (*Adaptive Situational Modes* – ASM). ASM'ler, durumlar ODD'nin sınırına ve dışına doğru ilerlediğinde hataların oluşmasını önlemek için kullanılan prosedürlerdir. Bu prosedürler en iyi şekilde örnekler aracılığıyla açıklanmaktadır.

Hava Trafik Kontrolörleri (*Air Traffic Controllers* – ATC) dünyadaki en kritik işlerden bazılarına sahiptir. Hava trafiğini farklı yüksekliklerde ve yerde yönlendirmenin yanı sıra programları sürdürmekten ve her şeyden önce güvenliği sağlamaktan sorumludurlar. Tek bir hata bile yüzlerce kişinin hayatına malolabilir. Değişen çevresel koşullar, trafik ve yorgunluk ortamında güvenliği sağlamanın bir yolu ASM'lerdir. ATC'ler, tehlikeli veya daha kolay hataya neden olan durumlar için önceden tanımlanmış çalışma modlarına sahiptir.

Yoğun trafik dönemlerinde, kontrolörler uçaklar arasında verimli mesafe bırakmaya öncelik verebilir ve normal ayrılma standartlarını düzenleyebilirler. Olumsuz hava koşullarında ATC'ler, kara trafiği yerine alternatif bir yaklaşım olan uçuşların güzergâh değiştirmesine ve bekleme düzenlerinin yönetilmesine odaklanabilirler. Eğer sistem kesintileri veya mekanik kesintiler meydana gelirse, ATC'lerin güvenliği sağlamak için geçiş yapabileceği ve alternatif iletişim yöntemlerine güvenebileceği uyarlanabilir kuralları vardır.

Hava Trafik Kontrolünde ve ASM'leri kullanan diğer kritik işlerde ve endüstrilerde daha birçok örnek bulunmaktadır. ASM'lerin en önemli çıkarımlarından biri, hataların mümkün ve hatta muhtemel olduğu inancıdır. ASM'ler normal çalışma modlarından, mevcut sistemin tasarım özelliklerini ve parametrelerini olmuş farz etmeyen özel modlara geçerler.

Fark etmiş olabileceğiniz gibi, iyi çözüm yolları neredeyse tamamen son savunma hattı olarak sistemin kullanıcılarına ve operatörlerine bağlıdır. İsviçre Peyniri Modeli ışığında, bu mantıklıdır. Gizli hatalar tespit edilemez ve önlenemez. Sistem ODD'den farklı bir durumdaysa, yapılabilecek tek şey operatörün aktif arızasını veya doğru davranışını beklemektir.

Kaynak: KLM.nl/

Gizli hatalar – Tasarımda miras ve gelecek

«Teknolojinin gelişiminde, en büyük tehlikelerin artık büyük bir bileşenin arızalanmasından veya izole operatör hatalarından değil, çoğunlukla organizasyonel ve yönetsel sektörlerde meydana gelen sonradan etkileyen insan hatalarının gizlice birikmesinden kaynaklandığı bir noktaya gelinmiştir.»

Reason, bu iddiayı 1990'da ortaya atıyor. Eğer söylediği o zaman geçerliyse, şimdi de kesinlikle geçerlidir. Ancak çabalarımız ne sıklıkla bu zorluğu kabul etmek şöyle dursun, çözmeye yönelik oldu. Durum temelli eğitim, simülasyon ve uyarlanabilir durumsal modlar gibi en iyi çözümler hâlâ yaygın olarak kullanılmıyor. Daha da saçma olanı, bunların neden kullanılmaması gerektiğine dair gerçek bir sebebin olmamasıdır.

Gizli hatalar yalnızca tasarımcıların ve mühendislerin hataları değildir. Aynı zamanda diğer şeylerin yanı sıra yönetim kararlarını, kültürel normları ve hatta müşteri beklentilerini de içerirler. Bununla birlikte, tasarımcılar, potansiyel arızaların sistem geneline yayılmasında hayati bir rol oynamaktadır. Tasarımcıların gizli hataları azaltmasının başlıca yolu, sistemler arasında basit, karmaşık olmayan bağlantılar ve ilişkiler kurmaktır. Ayrıca sistemlerde, özellikle son kullanıcının eylemleri ile sonuçlar arasında şeffaflığı da artırabilirler.

Tasarımcılar aynı zamanda gizli hataların etkisini ve riskini azaltmak için işlevler arası çalışabilmenin yollarının da farkında olmalıdır. Son kullanıcıları, mühendisleri ve paydaşları bu alandaki en iyi uygulamalardan bazıları hakkında bilgilendirirken aynı zamanda onlara yukarıda bahsedilen zayıf çözümler ve paradokslar hakkında da uyarılarda bulunmalıdırlar.

Reason'ın sözlerinden bu yana internetin ve akıllı telefonların teknolojik devrimlerine tanık olduk. Artık üretken yapay zekâ, otonom araçlar ve yeşil enerji gibi başka bir zirvedeyiz. Gizli hatalar, teknolojilerimiz daha karmaşık ve anlaşılmaz hale geldikçe daha fazla artarak yayılmaya devam edecektir. Doğal olarak var olan yerleşik patojenleri azaltan sistemler oluşturmak için çaba göstermeli ve kendimizi gelecekteki gizli hataların etkilerine hazırlamak için çokdisiplinli bir şekilde çalışmalıyız. ●

Skeomorfizm: Yapay zekânın sana ihtiyacı var

Juan Williman
bootcamp.uxdesign.cc
Çeviri: **Ayşe Dağıstanlı**

Skeomorfizmin yükselişi ve düşüşü

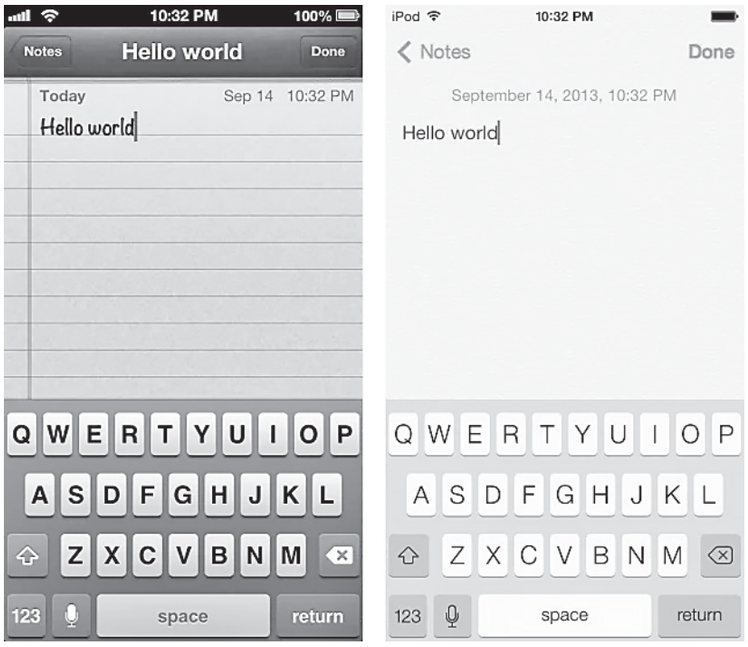
Skeomorfizm, ya da taklit nesneler başlangıçta kullanıcı arayüzlerine hakim olmuştu ve iOS da bunun unutulmaz bir örneğiydi (ancak ne Apple'a özgüydü ne de Apple tarafından icat edilmişti). Sanal dünyaya geçişi sorunsuz hale getirmek için gerçek yaşamdan aşina öğeleri kullanan iOS tasarımının ilk sürümleri skeuomorfizm konusunda tam bir ustalık örneğiydi. *Notlar* sarı not defterlerine benziyordu; *Newsstand*'ler ahşap raf görünümündeydi; *Game Center* cilalı ahşabı ve yeşil keçesiyle bir oyun masasını andırıyordu; düğmelerde derinlik ve gölgeler vardı ve bu da kullanıcılara dokunsal bir etkileşim hissi sağlıyordu.

Ancak dijital okuryazarlık arttıkça tasarımcılar düz tasarıma yöneldi. Google ve Apple'ın desteklediği bu minimalist yaklaşım, sadeliği ve verimliliği öne çıkarıyordu. Bu da, tasarımcıların çoğu kişinin işe yaramaz olarak gördüğü süs unsurlarını atmasına ve erişilebilirlik gibi önemli yönlere odaklanmasına olanak tanıdı. Böylece estetik açıdan hoş ve işlevsel açıdan etkili olan düz tasarım, sanal ve fiziksel dünyalar arasındaki boşluğu doldurmaya yardımcı olan *insani* unsurları ortadan kaldırdı.

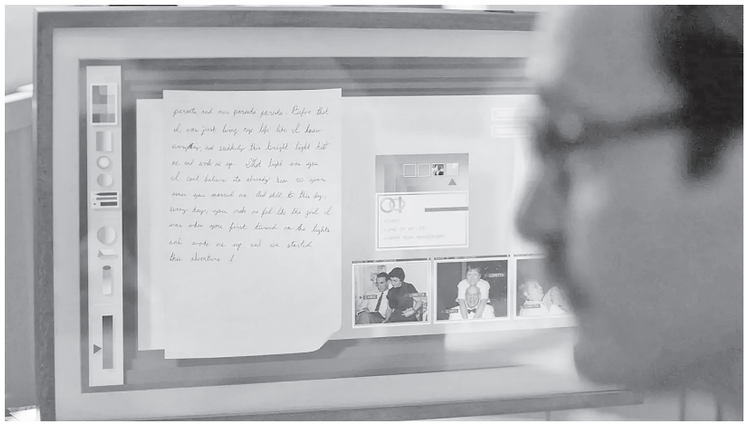
Yapay zekâ, teknolojiyle olan etkileşimlerimizde devrim yaratıyor, ama bu gelişmeler aynı zamanda kullanıcıları yabancılaştırabilecek soyutlamaları da beraberinde getiriyor. Yapay zekâ gelişmeye devam ettikçe, doğal ve sezgisel hissettiren arayüzlere duyulan ihtiyaç çok daha önemli hale geliyor. Ve *nesneleri taklit etme* sahneye giriyor. *Skeomorfik* tasarım, fiziksel dünyayı taklit eden unsurları yeniden dahil ederek, yapay zekâ odaklı arayüzleri daha insani ve hayatla ilişkili hale getirebilir. İşte bu yüzden *skeomorfizmin* bir geri dönüşe ihtiyacı var.

Yapay zekânın skeomorfizme ihtiyacı var

Yapay zekâ, gerekli girdileri veya etkileşimleri en aza indirirken kullanıcı için yarattığı değeri



Solda: iOS 6'daki Notes uygulaması; sağda: iOS 7'deki Notes uygulaması.



Her'de (2013) gösterilen ilk OS işletim sistemi.

en üst düzeye çıkararak benzeri görülmemiş bir şey başarıyor. UX tasarımı minimalizme doğru kaydığında bile, bilgisayarlar ve telefonlar, onlarla dokunmaya dayalı etkileşimlerimiz nedeniyle hâlâ biraz kişisel geliyor. Ancak yapay zekâ geliştikçe dokunma girdileri önemli ölçüde azalacak. İnsan sesi benzerleri bilgisayarların kişisel hissetmesini sağlayacak olsa da, görsel ipuçlarının da yeniden değerlendirilmesi gerekecek. Telefonunuzdaki asistanın insan taklidi sesi, nefes alıp vermeler, ses çatlamaları sessel etkileşim için bir tür skeuomorfizm olarak düşünülebilir. Amaç, sanal bir şeyin gerçekmiş gibi hissedilmesidir, tersi değil. Yapay zekânın, tıpkı akıllı telefonların ilk piyasaya çıktığında ihtiyaç duyulduğu gibi, görsel ipuçları için skeomorfizm uygulaması gerekecek. Bu durumu destekleyen ana noktalar şunlardır:

Duygusal bağ

Fiziksel nesnelerle doğal bir duygusal bağımız vardır. Skeomorfizm, gerçek dünyadaki öğelerin görünüşünü ve dokunuşunu kopyalayarak bu duyguları uyandırabilir, aşinalık ve rahatlık hissi yaratabilir. Yapay zekâ hayatımızda daha yaygın hale geldikçe, gerçekçi tasarımlar yoluyla duygusal bir bağ kurmak kullanıcı memnuniyetini artırabilir ve akıllı telefonların ilk günlerinde olduğu gibi bilgisayarların yeniden kişisel ve tanıdık hissetmesini sağlayabilir.

Güven hissi yaratmak

Yapay zekânın karar verme süreçleri çoğu zaman anlaşılmaz ve gizemli görünebilir. Yapay zekâ sayesinde skeomorfik tasarım, metin ve görüntüleri yansıtan camın bulanık görünümünden ziyade, gerçek gibi hissettiren bir değer yaratabilir. Yapay zekânın karmaşıklıklarını ilişkilendirilebilir öğe katmanlarının arkasına saklamak, kullanıcılarda bilgisayarlarına güven duyma ve özelliklerini daha iyi kullanabilme hissini uyandırmaya yardımcı olacaktır.

Her (2013) filmi bu konseptin, kahramanın Samantha'yla değil ama ondan önce gösterilen "aptal" işletim sistemiyle ilişki kurmasına açık bir örnektir. Filmin başında Theodore (Joaquin Phoenix) bilgisayarına bir mektup yazdırmaktadır. Onunla bilgisayarı arasında fiziksel bir etkileşim yoktur, yalnızca bir ses vardır. Ancak ekrandaki sanal kâğıt Theodore'u o kadar bağlantılı ve kişisel hissettirir ki, sanki kendisi ile "el yazısı" mektubu arasında başka bir ortam, bir bilgisayar yoktur. Ve bu da, UX (Kullanıcı Deneyimi) tasarımı ve özellikle de skeomorfizm sayesindedir.

Yapay zekânın gizemlerini ilişkilendirilebilir unsurların katmanlarının arkasına saklayarak, sürekli yapay zekânın görevleri nasıl yerine getirdiğini düşünmekten kendimizi alıkoymuş oluruz. Bu, bilişsel yükü ortadan kaldırır ve ihtiyacımız olanı yapması için ona güvenmemizi sağlar. Ekranın bizi gerçek olduğuna inandırdığı dokulu bir kâğıt parçası üzerinde harflerin belirmesini izlemek, harflerin bulanık, akan bir dikdörtgen içindeki görüntüsünü izlemekten daha büyülü ve kişisel hissettirir. Aslında ekranın sihirli bir şekilde metni işlemediğini ve kâğıdın kendi kendine yazmadığını biliriz, ancak kâğıdın kendi kendine yazdığını gördüğümüzde onu anlıyor gibi hissederiz. Gizem yoktur, sadece sanat, iş, duygular ve değer var ve bilgisayarların yeniden bu şekilde hissetmesine ihtiyacımız var. ●

iOS 6'da skeuomorfik tasarımlar: Notlar, Newsstand ve Sesli Notlar.

YAZILAR
Grafik Tasarımcılar Meslek Kuruluşu Derneği adına sahibi
Zeynep Tonguç
Tasarım
Bülent Erkmen
Sorumlu Yayın Yönetmeni ve Tasarım Devamlılığı
Osman Tülü
Grafik Uygulama: Tipograf
Baskı: A4 Ofset
Ayda bir yayımlanır, para ile satılmaz.

Grafik Tasarımcılar Meslek Kuruluşu Derneği
Ortaklar Caddesi Bahçeler Sokağı 17/4
Mecidiyeköy 34394 İstanbul
Tel: (0212) 267 27 58
Faks: (0212) 267 27 59
info@gmk.org.tr www.gmk.org.tr