Emniyet Kritik Yazılım Geliştirme Projelerinde Karşılaşılabilecek Etik Problemler

M. Umut PİŞKEN

Savunma Teknolojileri ve Mühendislik A.Ş., Mühendislik ve Sertifikasyon Müdürlüğü, Ankara, Türkiye

mpisken@stm.com.tr

Özet. Hatalı çalışmaları durumunda can kaybına, ciddi yaralanmalara veya çevrenin ciddi zarar görmesine yol açabilecek olan sistemler emniyet kritik sistemler olarak sınıflandırılmaktadırlar. Bu sistemlerde kullanılan yazılımlar da emniyet kritik yazılım olarak nitelendirilmektedirler. Günümüzde emniyet kritik sistemlerde kullanılan yazılım miktarı geçmiş dönemlere göre ciddi şekilde artmıştır. Hava trafik kontrol birimleri, demiryolu sinyalizasyon sistemleri, nükleer santraller ve fly-by-wire sistemler gibi birçok emniyet kritik sistem, ciddi oranda yazılım içermektedir. Bu yazılımların ilgili geliştirme standartlarına uygun şekilde ve azami özen gösterilerek geliştirilmesi gerekmektedir. Ancak projelerin uymaları gereken takvim ve bütçe gibi bazı kısıtlar, projelerde emniyet konusunda bazı etik problemlerin yaşanmasına yol açabilmektedir. Dolayısıyla emniyet kritik yazılım geliştirme projelerinde çalışan mühendislerin karşılaşabilecekleri etik problemler konusunda bir farkındalıkları olması ve bu tarz durumlarla karşılaştıklarında nasıl bir hareket tarzı sergilemeleri gerektiğinin bilincinde olması gerekmektedir. Bu çalışmada emniyet kritik yazılım geliştirme projelerinde karşılaşılabilecek etik problemler incelenecek ve bu problemleri önlemeye yönelik öneriler sunulacaktır.

Anahtar Kelimeler. Emniyet Kritik Yazılımlar, Etik, Etik Problemler

1 Giriş

Hatalı çalışmaları durumunda can kaybına, ciddi yaralanmalara veya çevrenin ciddi zarar görmesine yol açabilecek olan sistemler emniyet kritik sistemler olarak sınıflandırılmaktadırlar. Bu sistemlerde kullanılan yazılımlar da emniyet kritik yazılım olarak nitelendirilmektedirler. Günümüzde emniyet kritik sistemlerde kullanılan yazılım miktarı geçmiş dönemlere göre ciddi şekilde artmıştır. Aviyonik sistemlerden örnek vermek gerekirse, 1987'de üretilen Airbus A320 uçağında yaklaşık olarak 800.000 satır kod mevcutken, 1994'te üretilen Boeing 777 uçağında yaklaşık 4 milyon satır kod bulunmaktadır [1]. 2009 yılında üretilen Boeing 787 uçağında ise yaklaşık 7 milyon satır kod bulunmaktadır [2]. Bu verilerden anlaşılacağı

üzere emniyet kritik sistemlerde kullanılan yazılım miktarı gittikçe yükselmektedir. Aviyonik sektörü özelinde, her 10 yılda bir, aviyonik sistemlerde kullanılan yazılım miktarının ikiye katlandığı öne sürülmektedir [1].

Yazılım hataları, yazılım geliştirme esnasında ortaya çıkan sistematik hatalardan kaynaklanmaktadırlar [3]. Yazılım hataları, fiziksel parçalarda görülebilen ve kullanıma bağlı yıpranma-aşınma kaynaklı oluşan rastsal hatalardan(Random Failure) farklıdırlar. Emniyet kritik yazılım geliştirme standartları da bu durumu göz önüne alarak, sistematik hataları önlemek üzere yazılım geliştirme süreçlerine yönelik belirli gereksinim ve kısıtlamaları tanımlayacak şekilde geliştirilmiştir [3]. Bu yaklaşıma göre, öncelikle sistem emniyet analizleri yapılarak yazılımın içinde çalışacağı sistem açısından emniyet kritiklik seviyesi belirlenmekte, sonrasında ise bu kritiklik seviyesini sağlamak üzere kullanılacak olan standardın istediği yazılım yaşamdöngüsü süreçleri işletilmektedir.

Emniyet kritik yazılımların ilgili standardın istediği yazılım yaşamdöngüsü süreçlerine göre geliştirilip geliştirilmediğine ilişkin kontrol sertifikasyon otoriteleri tarafından gerçekleştirilmektedir. Ancak projelerin büyüklükleri sebebiyle, bu kontroller proje büyüklüklerine göre görece küçük örneklemler kullanılarak ve projede çalışan mühendisler ile mülakatlar yapılarak gerçekleştirilmektedir. Dolayısıyla bu süreçte projede çalışan mühendislerin iyi niyet sergilemesi ve sertifikasyon otoritesi ile işbirliği sağlaması büyük önem arz etmektedir. Aksi halde, zaman kısıtı ve örneklem boyutu sebebiyle, kontrolü gerçekleştiren otoritenin gözünden bazı eksiklikler kaçabilir ve yazılım ilgili standartların istediği tüm yazılım yaşamdöngüsü amaçlarını sağlamadığı halde başarılı şekilde kontrollerden geçebilir. Bu tarz bir durumda ortaya çıkacak yazılım ürünü kullanıldığı sistem içerisinde hatalara sebebiyet vererek can kaybı, yaralanma veya çevrenin ciddi zarar görmesi gibi istenmeyen olaylara yol açabilecektir. Bu sebeple, emniyet kritik yazılım geliştiren ekiplerde görev alan mühendislerin etik sorumluluklarının farkında olması ve sadece çalıştıkları firmanın çıkarlarını değil, toplumun genelinin de faydasını göz önüne almaları gereklidir.

Bu çalışmanın ikinci kısmında emniyet kritik yazılımlar ve bu yazılımların geliştirilmesine ilişkin kullanılan standartlar hakkında genel bilgiler verilecektir. Çalışmanın üçüncü kısmında etik ve etik problemlerden bahsedildikten sonra, dördüncü kısımda emniyet kritik yazılım geliştirme projelerinde karşılaşılabilecek etik problemler incelenecektir. Çalışmanın son bölümünde ise sonuç ve önerilere yer verilecektir.

2 Emniyet Kritik Yazılımlar

Emniyet kritik sistemlerin bir parçası olan ve hatalı çalışmaları durumunda sistem emniyetine etki eden yazılımlar emniyet kritik yazılımlar olarak adlandırılmaktadır [4]. Yazılım insanlığın ürettiği en karmaşık ürünlerden birisi olarak sınıflandırılmaktadır [4]. Özellikle büyük sistemlerde yer alan yazılımları olası bütün girdileri deneyerek test etmek olanaklı olmamakta, bu sebeple ne kadar test yapılırsa yapılsın yazılımın hata içermediği garanti altına alınamamaktadır [4]. Günümüzde

emniyet kritik sistemlerde kullanılan yazılım miktarı da gittikçe artmakta, bu sebeple üretilen yazılımların istenilen emniyet seviyesini sağladığının garanti altına alınması daha da önem kazanmaktadır. Mekanik sistemler için gerçekleştirilen sistem hata ve güvenilirlik analizleri ne yazık ki karmaşık yapısından ötürü yazılımlara uygulanamamaktadır [4]. Yazılım, mekanik sistemlerden farklı olarak fizik yasalarına göre hareket etmemekte, dolayısıyla yazılımın hangi koşullar altında hatalı çalışacağı tahmin edilememektedir [4].

Yazılımların güvenirliği testler veya güvenilirlik analizleri ile tespit etmek mümkün olmadığı için, tasarım teminatı kavramı ortaya atılmıştır [4]. Tasarım teminatı ile gereksinim, tasarım ve geliştirme aşamalarında ortaya çıkabilecek hataların sistem emniyeti açısından kabul edilebilir seviyeye indirildiğini garanti altına almak için uygulanan planlı ve sistematik aktiviteler kastedilmektedir[5]. Tasarım teminatı yaklaşımı, yazılım geliştirme esnasında uygulanacak süreçler ne kadar titiz ve sıkı olursa yazılımda o derece az hata kalacağı varsayımına dayanmaktadır[4]. Bu yaklaşımda, yazılımın kritiklik seviyesi yükseldikçe, uygulanan geliştirme ve doğrulama aktiviteleri nicelik ve nitelik açısından artmaktadır.

Emniyet kritik yazılımlar için, farklı alanlarda farklı geliştirme standartları mevcuttur. Aşağıdaki tabloda bu standartlardan bazıları ve kullanıldıkları alanlar verilmiştir:

Standart Adı	Kullanıldığı Alan
EN 50128 - Railway Applications: Software for Railway Control and Protection Systems	Hızlı Tren, Demiryolu Kontrol Yazılımları
DO-178C - Software Considerations in Airborne Systems and Equipment Certification	Aviyonik Yazılımlar
IEC 62138 Nuclear Power Plants - Software Aspects for Computer-based Systems Performing Category B or C functions	Nükleer Santral Yazılımları
IEC 62304 Medical Device Software – Software Life Cycle Processes	Medikal Cihazlardaki Yazılımlar

Tablo 1. Emniyet Kritik Yazılım Geliştirme Standartları

Emniyet kritik yazılım geliştirme standartlarında, sistem seviyesi emniyet analizlerinden gelen yazılım tasarım teminatı seviyelerine ilişkin yazılım yaşamdöngüsü boyunca karşılanması gereken amaçlar belirlenir [4]. Örnek vermek gerekirse, DO-178C standardında tasarım teminatı seviyesi DAL A olarak belirlenmiş olan yazılımlar için, geliştirilen kodun yazılım tasarımı ile uyumlu olup olmadığının kontrolünün bağımsız bir şekilde (kod geliştiricilerden farklı kişiler tarafından) yapılması gerekirken, tasarım teminatı seviyesi DAL B olarak belirlenen yazılımlarda

bu işlemin bağımsız şekilde yapılmasına gerek yoktur, bir başka ifadeyle kodu geliştiren kişilerin bu kontrolü yapması yeterlidir[6]. Benzer şekilde EN 50128 standardına göre tasarım teminatı seviyesi SIL 3 ve SIL 4 olan yazılımlar için sınır değer testlerinin yapılması gerekirken, SIL 1 ve SIL 2 olan yazılımlar için yapılması şart değildir [7].

3 Etik

Etik, bir kişinin ya da grubun davranışlarına rehberlik eden doğrunun veya yanlışın, iyi veya kötünün standartlarını oluşturan moral ilkelerin kodu olarak tanımlanmaktadır[8]. Etiğin amacı çeşitli eylem alternatifleri arasında bir seçim yapmada insanlara yardımcı olan davranış ilkelerini oluşturmaktır [8]. Etik, teorik etik ve uygulamalı etik olarak iki ana bölümde sınıflandırılmaktadır [9]. Teorik etik daha kuramsal ve genellik taşıyan bir yapıya sahipken, 20. yüzyılın son çeyreğinde, dünyada hayatın her alanında giderek artan somut ahlâki problemlerin bir sonucu olarak ortaya çıkıp birtakım münferit konularda problem çözmeyi ve dolayısıyla vizyonumuzu geliştirmeyi amaçlayan uygulamalı etik için bu durum geçerli değildir[9]. Meslek etikleri, uygulamalı etik kapsamına girmektedir. Meslek etiği, belirli bir meslek grubunun, mesleğe ilişkin olarak oluşturup koruduğu, meslek üyelerine emreden ve onları belirli bir şekilde davranmaya zorlayan, kişisel eğilimlerini sınırlayan, yetersiz ve ilkesiz üyeleri meslekten dışlayan, meslek içi rekabeti düzenleyen ve hizmet ideallerini korumayı amaçlayan mesleki ilkeler bütünüdür [8]. Mesleki etik kapsamında ilke ve standartlar belirlenirken dayanak noktası olarak etik kuramlar kullanılmaktadır[8]. Bu kuramlardan başlıcaları şunlardır:

- **Teleolojik Yaklaşım**: Bu yaklaşıma göre bir eylem kendi başına iyi veya kötü değildir, o eylemi iyi ya da kötü yapan şey sonuçlarıdır. Dolayısıyla bir eylem istenen sonucu ortaya çıkarıyorsa, o zaman ahlaki açıdan doğru ve iyi kabul edilmelidir. Faydacı anlayış da bu yaklaşıma girmektedir. Faydacı anlayışa göre, bir eylemin ahlaki bakımdan doğru sayılabilmesi için eylemin sağladığı toplam faydanın, bireyin yapacağı başka bir eylemin sağlayacağı toplam faydadan büyük olması gerekir.
- **Deontolojik Yaklaşım**: Bu yaklaşıma göre eylemler, eyleme temel olan, eylemi ortaya koyan düşünüşün niteliğine göre değerlidir veya değildir. Ödev etiği olarak da adlandırılmaktadır. Eylemlerin ahlaki olup olmadığı eylemin sonucunda değil niyetinde aranmalıdır.

Etik dışı davranış, bireylerin, grupların veya örgütlerin, toplumun ve/veya mesleğin iyi, güzel ve doğru saydığı etik kuralları, ilkeleri terk etmesi ve bu kural ile ilkelere aykırı davranış göstermesidir [8]. Etik dışı davranışın bireysel nedenleri olarak aşağıdaki unsurlar sıralanabilir:

- Etik ikilemler
- Etik standartlardaki farklılıklar

- Bencil davranma
- Mesleki bilgi yetersizliği

Etik ikilemler: Etik ikilemler, yöneticilerin veya işgörenlerin karar vermesini engelleyen, net olmayan karmaşık durumlardır [8]. Kişi farklı alternatifler arasında karar vermek zorunda olduğunda, bir etik çıkmaz oluşmaktadır [8].

Etik standartlardaki farklılıklar: Bireyler, toplumca benimsenen değerleri öncelikle ailelerinden, daha sonra ise okuldan öğrenmeye başlamaktadırlar. Farklı aile yapıları ve eğitim olanakları, kişilerin etik standartlarının farklılaşmasına neden olabilmektedir [8]. Bireyler etik dışı davranışlarını rasyonelleştirmek için aşağıdaki gibi birtakım savunma mekanizmaları geliştirebilirler [10].

- Herkes yapıyor
- Eğer işimi kaybedersem kimseye yararı olmaz

Bencillik: Bencillik kişinin insancıl eğilimlerini ve duygularını geriye iterek, sadece çevresini kendi çıkarlarına uygun bir sömürü ortamına dönüştürmeye yönelik bir çıkar güdüsü olarak ifade edilebilir [11]. Kişiler yaptıkları davranışın uygunsuz olduğunu bilseler bile, çıkarları için bu davranışı sürdürebilirler [8].

Mesleki yetersizlik: Gerek çalışan, gerekse yöneticilerin, çalıştıkları örgütte, mesleğini yerine getirebilmek için birtakım yetkinliklere sahip olması gerekir. Bireyler gerekli mesleki bilgi, tecrübe ve yeterliliğe sahip olmadığında hem nicelik hem de nitelik olarak işin gereğini yerine getiremezler.

4 Emniyet Kritik Yazılım Geliştirme Projelerinde Karşılaşılabilecek Etik Problemler

Emniyet kritik yazılımlar, genellikle bir sertifikasyon otoritesi veya belgelendirme yetkisine sahip bir kuruluş tarafından değerlendirilmektedir [12]. Bu değerlendirme neticesinde ilgili yazılımın gerekli emniyet seviyesini sağlayıp sağlamadığına ilişkin karar verilmektedir. Tasarım teminatı yaklaşımı sebebiyle, bu değerlendirmeler ürün odaklı olmaktan ziyade, süreç odaklı olmakta ve emniyet kritik yazılım geliştirme yaşamdöngüsü süreçleri incelenmektedir [12]. Bu incelemeler esnasında proje çalışanları ile mülakatlar yapılmakta, yaşamdöngüsü boyunca karşılanması gereken amaçlara yönelik olarak çıktılar incelenmektedir. Ancak yazılım projelerinin büyüklükleri sebebiyle, tüm çıktıların incelenmesi zaman ve kaynak bakımından mümkün olamadığı için örneklemlere bakılmaktadır. Dolayısıyla değerlendirmeyi yapan otoritelerin yaşamdöngüsü süreçlerinden olabilecek olası sapmaların tamamını tespit etmeleri her zaman mümkün olamamaktadır.

Bu noktada emniyet kritik yazılım geliştiren ekiplerin etik açıdan uygulanması gereken standardı benimsemeleri ve sadece otorite istediği için değil, yazılımın gerçekten emniyetli olması ve kullanım esnasında can kaybı veya ciddi yaralanma gibi istenmeyen olaylara sebebiyet vermemesi için gereken tüm çabayı göstermeleri gerekmektedir. Ancak ne yazık ki, emniyet kritik yazılım geliştirme projelerinde

verilen kararların birçoğu emniyet odaklı olmaktan öte ekonomik odaklı olmaktadır [13].

Konuya ilişkin olarak, radyoterapi tedavilerinde kullanılan ve yazılım içeren Therac-25 cihazının kullanımı esnasında yaşanmış olan istenmeyen olay örnek verilebilir. Therac-25 cihazı içerdiği yazılım hatasından dolayı dört kişinin radyoterapi esnasında doz aşımı nedeniyle ölümüne yol açmıştır [14]. Yapılan incelemelerde kazaya sebep olan Therac-25 cihazının içindeki yazılım geliştirilirken, temel yazılım mühendisliği prensiplerine uyulmadığı, testlerin yetersiz seviyede olduğu, gerekli dokümantasyonun üretilmediği ve kalite güvence süreçlerine uyulmadığı tespit edilmiştir [15].

Emniyet kritik yazılım projelerinde, takvim ve bütçe tahminlerinde yapılacak olası hatalar, projelerin yaşamdöngüsü esnasında zaman baskısı ve kaynak eksikliğine sebebiyet vermekte, bu durum ise yöneticiler ve çalışanlar açısından etik ikilemlerin doğmasına yol açmaktadır. Zaman baskısı sebebiyle, ilgili geliştirme standardında istenilen amaçlar beklenen seviyede gerçekleştirilemeyebilmektedir. Dodd ve Habli tarafından yapılan çalışmada, havacılık sektöründe sertifikasyon otoritesi adına denetlemeleri gerçekleştiren kişilerle yapılan mülakatlarda, firmaların denetime hazır olmadıkları halde, denetleme talep etmeleri ve denetimden başarılı şekilde geçmeyi umuyor olmaları bir problem olarak ifade edilmiştir [3]. Bu durum zaman baskısından kaynaklı etik problemlere güzel bir örnek teşkil etmektedir. Firmada çalışan mühendisler, denetime hazır olmadıklarının farkında olmalarına rağmen zaman baskısı sebebiyle denetime girip geçmeyi ummaktadırlar. Denetimlerin amacı emniyet kritik yazılımların beklenen tasarım teminatı seviyesini karşıladığının garanti altına alınmasıdır. Bu sebeple firmaların ve mühendislerin etik açıdan, topluma karşı olan sorumlulukları gereği hazır olmadıkları halde sertifikasyon denetimlerine girmeyi talep etmeleri uygun değildir.

Projelerde gene zaman baskısı sebebiyle ilgili standartların istediği aktiviteler istenilen nitelikte gerçekleştirilemeyebilmektedir. Örneğin, gözden geçirme süreçleri, zaman baskısı sebebiyle, gözden geçirme katılımcılarına yeterli süre tanınmadan gerçekleştirilebilmektedir. Bu tarz durumlarda, gözden geçirmeye konu olan yazılım yaşamdöngüsü çıktıları (gereksinim, tasarım, kod vs.) standardın istediği seviyede incelenmemekte ve hatalar tam olarak giderilmemektedir. Bu durum da etik problemlere örnek olarak gösterilebilir. Mühendislerin, kendilerine işlerini standardın istediği seviyede yapamayacakları bir süre verildiği zaman, duruma itiraz etmeleri, ek süre istemeleri, verilmediği takdirde ilgili aktiviteye katılmamaları veya konuyu ilgili otoriteye iletmeleri etik bir sorumluluktur.

Emniyet kritik yazılım geliştirme projelerinde kullanılacak yöntem ve araç seçimlerinde de mutlaka yapılacak seçimin yazılım emniyetine olan etkisi göz önüne alınmalıdır. Örneğin nesne yönelimli programlama dillerinin uygun olmayan şekilde kullanımı yazılımın deterministik olmayan davranışlar sergilemesine ve sistem emniyetini tehlikeye sokmasına sebep olabilmektedir [13]. Bu sebeple, kullanılacak programlama dili seçilirken, karara katılan mühendisler mutlaka bu durumun yazılım emniyetine yapabileceği olası olumsuz etkileri de göz önüne almalıdırlar. Bu tarz seçimlerin, sadece geçmiş kullanım alışkanlıklarına veya firmanın mevcut yazılım

geliştirme ortamına bakılarak yapılması firma açısından verimlilik sağlıyor gibi gözükse de etik açıdan uygun olmayacaktır.

Benzer şekilde, formal yöntemler gibi yazılım sektöründe çok yaygın olarak kullanılmamasına rağmen yazılım emniyetine ciddi şekilde katkı sağlayan ve standartlar tarafından kullanımları tariflenen yeni yöntemlerin de mühendisler ve firmalar tarafından benimsenmesi ve uygulanması etik bir sorumluluktur. Bu tarz yeni yöntemlere karşı, mühendisler ve firmalar bilgi eksikliği sebebiyle sıcak bakmamakta, öğrenme süresinin proje takvimine olumsuz yansıyabileceği ve konuya ilişkin eğitim/danışmanlık gibi ek masraflar sebebiyle bütçe üzerinde de olumsuz etkileri olacağını düşünmektedirler[13]. Konuya etik açıdan bakıldığında ise, bu yöntemler yazılım emniyetine katkı sağladığı için, yol açtıkları masraftan bağımsız olarak uygulanmaya çalışılmalıdır.

Emniyet kritik yazılım geliştirme projelerinde çalışan mühendislerin, projede uygulanacak standarda hâkim olmaları gerekmektedir. Standarda hâkim olmamaktan kaynaklanacak problemler de mühendislerin etik açıdan sorumluluğuna girmektedir [14]. Projede uygulanacak olan standarda tam olarak hâkim olmayan mühendisler, standardın istediği yazılım yaşamdöngüsü amaçlarını yanlış yorumlayabilir ve tam olarak gerçekleştiremeyebilirler. Bu tarz durumlarda, konuyla ilgili yeteri kadar bilgim yoktu şeklindeki mazeretler etik açıdan sorumluluğu ortadan kaldırmamaktadır. Mühendislerin uygulanacak standarda ilişkin her türlü bilgiyi edinmeye çalışmaları ve gerekiyorsa çalıştıkları firmadan bu konuda eğitim almayı talep etmeleri gerekmektedir. Firmaların da gerekli eğitimleri mühendislerine sağlamaları etik açıdan bir gerekliliktir.

Son olarak emniyet kritik yazılım geliştirme projelerinde, mühendislik kararlar alınırken emniyete olan etkilerin performans, kullanılabilirlik veya ekonomik etkilerden daha öncelikli olarak göz önüne alınması etik açıdan bir gerekliliktir. Kullanılabilirliği veya performansı yükseltmek adına sistem emniyetinden taviz vermek etik açıdan uygun değildir.

5 Sonuç ve Öneriler

Bu çalışmada, emniyet kritik yazılım geliştirme projelerinde karşılaşılabilecek etik problemler ortaya konulmaya çalışılmıştır. Günümüzde emniyet kritik sistemlerde kullanılan yazılım miktarı gittikçe arttığı için, bu sistemlerdeki yazılımların can kaybı, ciddi yaralanma gibi istenmeyen olaylara sebebiyet verme olasılığı da yükselmektedir. Bundan dolayı, diğer yazılım geliştirme projelerinden farklı olarak, emniyet kritik yazılım geliştirme projelerinde etik konularda daha titiz davranılması gerekmektedir. Mühendislerin karşılaşabilecekleri etik problemlerin farkında olmaları, bu tarz problemlerle projelerin yaşamdöngüsü esnasında karşılaştıklarında, daha hızlı ve etik açıdan doğru kararlar vermelerine katkı sağlayacaktır. Etik konularda farkındalığı arttırmak adına, üniversitelerin mühendislik bölümlerinde meslek etiğine yönelik dersler eklenmesi faydalı olacaktır.

Firmaların emniyet kritik yazılım geliştirme projelerinde çalışan yöneticilere ve mühendislere rehberlik etmesi açısından, etik kodlar belirlemeleri gerekmektedir.

Böylece, proje esnasında karşılaşılabilecek etik problemlerin çözümüne ilişkin izlenecek genel politikalar belirlenmiş olacaktır. Çalışanlar bu politikalara ve etik kurallara dayanarak belirli noktalarda itirazlarını daha rahat dile getirebilecekler, bunun sonucunda da etik açıdan doğru kararların alınması mümkün olacaktır.

Firmalar emniyet kritik yazılım geliştirme projelerinde çalışan yönetici ve mühendislerin etik sorumlulukları konusunda farkındalıklarını arttırmak amacıyla, düzenli periyotlarla firma içi eğitimler düzenleyebilirler. Bu hem çalışanların etik konusundaki farkındalığını arttıracak, hem de çalışanlara firmanın etik sorumluluklar konusuna önem verdiği konusunda net bir mesaj iletecektir.

Emniyet kritik yazılım geliştiren firmaların emniyet kültürü geliştirmeleri ve bunu firma geneline yaymaları etik problemlerin daha oluşmadan önlenmesini sağlayabilir. Emniyet kültürü ile, proje esnasında alınacak her türlü kararın diğer tüm etmenlerden öncelikli olarak emniyet açısından bakılarak alınması anlatılmak istenilmektedir. Bu sayede önceki bölümlerde örnekleri verilen etik problemlerin oluşması önlenmiş olacaktır.

Emniyet kritik yazılımlar, her ne kadar belirlenen standartlara uygun şekilde geliştirilseler de, hata içerebilirler. Burada önemli olan nokta, geliştirme esnasında firmaların ve mühendislerin etik açıdan doğru hareket etmeleri ve ellerinden gelen en iyi çabayı ortaya koymalarıdır. Aksi halde, oluşabilecek istenmeyen kazalarda firmaların ve mühendislerin etik olmayan davranışlarının da payı olacaktır.

Gelecekte bu konuya ilişkin olarak, emniyet kritik yazılım geliştirme projelerinde çalışan mühendislerin etik algılarını ölçmeye yönelik saha çalışmalarının yapılması faydalı olacaktır. Böylece konuya ilişkin problemler daha net olarak ortaya konulabilir. Ayrıca emniyet kritik yazılım geliştirme projelerine özel bir etik kod setinin belirlenmesi de, bu konuda çalışan mühendis ve firmalara yol göstermek adına faydalı olacaktır.

Kaynaklar

1. Ian Moir, Allan G. Seabridge, “Civil Avionics Systems” John Wiley & Sons, 2nd Edition, 2013.
2. Burak Ata, Özgür Babur, “Emniyet Kritik Aviyonik Yazılımlarda Tespit Edilememiş Hataların Yol Açtığı İstenmeyen Durumlar ve Kazalar”, 5. Ulusal Yazılım Mühendisliği Sempozyumu 2011 Bildiriler Kitabı, Sayfa 149-152, 26-28 Eylül 2011.
3. Ian Dodd, Ibrahim Habli, “Safety Certification of Airborne Software:An Empirical Study”, Reliability Engineering and System Safety, 98, Sayfa 7–23, 2012.
4. Leanna Rierson, “Developing Safety-Critical Software”, CRC Press, 1st Edition, 2013.
5. SAE ARP4754A, “Guidelines for Development of Civil Aircraft and Systems”, 2010.
6. RTCA, “DO-178C Software Considerations in Airborne Systems and Equipment Certification”, 2011.

7. CENELEC, “EN 50128 - Railway applications: software for railway control and protection systems”, 2001.
8. Alptekin Sökmen, Serdar Tarakçıoğlu, “Mesleki Etik”, Detay Yayıncılık, 1. Baskı, 2013.
9. Ahmet Cevizci, “Felsefe”, Anadolu Üniversitesi Yayınları, 1. Baskı, 2012.
10. Çiğdem Kırel, “Örgütlerde etik davranışlar, yönetimi ve bir uygulama çalışması, Anadolu Üniversitesi Yayınları, 2000.
11. Adil İzveren, “ Toplumsal törebilim: sosyal ahlâk”, Ankara İktisadi ve Ticari İlimler Akademisi Yayınları, 1980.
12. Sunil Nair, Jose Luis de la Vara, Mehrdad Sabetzadeh, Lionel Briand, “An Extended Systematic Literature Review on Provision of Evidence for Safety Certification”, Information and Software Technology Vol. 56, Sayfa 689–717, 2014.
13. Jonathan Bowen, “The Ethics Of Safety-Critical Systems”, Communications Of The Acm, Vol. 43, No. 4, Sayfa 91-97, 2000.
14. Douglas Birsch, “Moral Responsibility for Harm Caused by Computer System Failures”, Ethics and Information Technology, Vol. 6, Sayfa 233–245, 2004.
15. Nancy G. Leveson, Clark S. Turner, “An Investigation of the Therac-25 Accidents”, Vol. 26, No. 7, Sayfa 18-41, 1993.