**YAZILIM KALİTESİ**

Yazılım kalitesinin SWEBOK rehberinin bilgi alanlarına (KA) dahil edilmesinin nedenlerinden biri yazılımda iki anlama gelmesidir. Bunlar: yazılım ürünlerinin istenilen özellikleri, yazılım ürününün bu özellikleri elde etmek için kullandığı süreçler, araçlar ve teknikler olarak belirtilebilir. Bazı yazarlar ve kuruluşlar kaliteyi çok farklı tanımlamıştır ama genel olarak varmak istedikleri nokta; kalite ile yazılım ürününün gereksinimlere uygunluğu arasında güçlü bir bağlantının olduğudur. Yazılım kalitesi gereksinimleri aslında işlevsel gereksinimlerin öznitelikleri olarak tanımlanabilir.

**1. Yazılım Kalitesi Temelleri**

Tüm paydaşlar için kaliteyi neyin oluşturduğu konusunda anlaşmaya varmak ve bu anlaşmanın yazılım mühendislerine açıkça iletilmesi, kalitenin birçok yönünün resmi olarak tanımlanmasını ve tartışılmasını gerektirir. Yazılım gereksinimleri, yazılımın ve ilgili belgelerin istenilen kalite seviyelerine ulaşma derecesini değerlendirmek için ölçüm yöntemlerini ve kabul kriterlerini etkiler.

**1.1. Yazılım Mühendisliği Kültürü ve Etiği**

Sağlıklı bir yazılım mühendisliği kültürü, maliyet, zamanlama ve kalite arasında denge sağlar. Güçlü bir yazılım mühendisliği etiği, mühendisliği kaliteyle ilgili bilgileri, koşulları ve sonuçları doğru bir şekilde rapor ettiğini varsayar. Etik yazılım kalitesinde, kültürde ve yazılım mühendislerinin tutumlarında da önemli bir rol oynar.

**1.2. Kalitenin Değeri ve Maliyetleri**

Yazılım kalitesi düzeyini belirlemeye yardımcı olmak için, yazılım kalitesi maliyeti (CoSQ) ölçümü kullanılır. Düşük kalite, yazılım ürününün belirtilen gereksinimleri tam olarak karşılamadığı anlamına gelir. Dört kalite kategorisi maliyeti vardır: önleme, değerlendirme, iç arıza ve dış başarısızlık. Önleme yazılım süreci, kaliteli altyapı, kalite araçları, eğitim, denetimler ve yönetim değerlendirmelerini içerir. Değerlendirme maliyetleri genellikle bir projeye özgü değildir; yapılan organizasyon, kusurları olan proje faaliyetlerinden kaynaklanır. Bu değerlendirme faaliyetleri değerlendirme maliyetleri ve test maliyetleri olarak sınıflandırılabilir. Değerlendirme maliyetleri, değerlendirme faaliyetleri sırasında bulunan ve yazılım ürününün müşteriye tesliminden önce keşfedilen hatalardır. Dış arıza maliyetleri, yazılım ürünü müşteriye teslim edildikten sonra keşfedilen yazılım sorunlarına yanıt verme etkinliklerini içerir.

**1.3. Modeller ve Kalite Özellikleri**

Yazılım kalitesi özellikleri için çeşitli yazarlar, yazılım ürünlerinin kalitesini tartışmak, planlamak ve derecelendirmek için yararlı olabilecek yazılım kalitesi özellik modelleri üretmiştir. ISO/IEC 25010: 2011 kalite ve kullanımda ürün kalitesi olmak üzere iki kalite modeli olarak tanımlar. SWEBOK Kılavuzu'ndaki Ek B, her KA için geçerli standartların bir listesini sağlar. Bu KA standartları, yazılım kalitesini nitelemenin çeşitli yollarını kapsamaktadır.

**1.3.1. Yazılım Süreç Kalitesi**

Yazılım kalite yönetimi ve yazılım mühendisliği süreç kalitesi, yazılım kalitesi üzerinde doğrudan bir etkiye sahiptir. Yazılım kuruluşlarının yeteneklerini değerlendiren modellerin öncelikleri proje organizasyonu ve yönetim konularıdır, süreç sonuçları ürünleri içerdiğinden, süreç kalitesini ürün kalitesinden tamamen ayırmak mümkün değildir. Bir kurumun sürekli olarak istenilen kalitede ürünler üretme yeteneğine sahip olup olmadığını belirlemek kolay değildir.

**1.3.2. Yazılım Ürün Kalitesi**

Yazılım mühendisleri, başta kalite gerekliliklerini ortaya çıkarmak ve bunlara ulaşmadaki zorluk düzeylerini anlamakla yükümlüdürler. Tüm yazılım geliştirme süreçleri kalite gereklilikleri göz önünde bulundurularak tasarlanır, güvenlik ve güvenilirlik gibi nitelikler önemliyse ek geliştirme maliyetleri taşıyabilir. Ek geliştirme maliyetleri, elde edilen kalitenin beklenen faydalara karşı takas edilmesini sağlamaya yardımcı olur. Kalite ile ilgili ara iş ürünlerinin yazılım mühendisliği süreci boyunca değerlendirilmesi gerekir.

**1.4. Yazılım Kalitesinin İyileştirilmesi**

Yazılım ürünlerinin kalitesi, yinelemeli bir süreç yoluyla geliştirilebilir: (1) yazılım yaşam döngüsü süreçleri, (2) hata algılama, kaldırma ve önleme süreci, (3) kalite iyileştirme süreci. Kusurların önlenmesi ve erken teşhisi yoluyla kalitenin geliştirilmesi ve sürekli iyileştirme kalite iyileştirmenin arkasındaki teori ve kavramlardır. Bu kavramlar, bir ürünün kalitesinin, onu oluşturmak için kullanılan sürecin kalitesiyle doğrudan bağlantılı olduğunu belirtir.

**1.5. Yazılım Güvenliği**

Güvenlik açısından kritik sistemler, sistem arızasının insan hayatına, diğer canlılara, fiziksel yapılara veya çevreye zarar verebileceği sistemlerdir. Örneğin toplu taşıma sistemleri ve tıbbi cihazlar. DO-178C [1] gibi endüstri standartları ve kritik güvenlik yazılımı geliştirmek için yeni ortaya çıkan süreçler ve teknikler vardır. Bu standartların ve tekniklerin amacı, yazılımda hata riskini azaltmak ve böylece yazılım güvenilirliğini artırmaktır.

**2. Yazılım Kalite Yönetim Süreçleri**

Yazılım kalite yönetimi, tüm yazılım ömrü boyunca süreçleri, çıktılarını, ölçümlerini ve geri bildirimleri tanımlar. Dört alt kategoriden oluşur: yazılım kalite planlaması, yazılım kalite güvencesi, yazılım kalite kontrolü ve yazılım süreci geliştirme. Yazılım kalite planlamasını tahmin etmeyi içerir. Yazılım Kalite Güvencesi yazılım süreçlerinin amaçlarına uygun olduğuna dair kanıtlar sağlamak için yazılım süreçlerinin yeterliliğini tanımlar ve değerlendirir. Yazılım Kalite Kontrolü, proje için belirlenen standartlara uyup uymadıklarını belirlemek için belirli proje yapılarını inceler, ara ürünlerin yanı sıra nihai ürünleri de değerlendirir. Yazılım Süreci Geliştirme, yazılım kalitesini artırmak amacıyla süreç etkinliğini, verimliliğini ve diğer özelliklerini geliştirmeyi amaçlar. Yazılım planlarının nasıl uygulandığını, ara ve nihai ürünlerin belirtilen gereksinimleri ne kadar iyi karşılandığını gösteren görev ve tekniklerden oluşur.

**2.1. Yazılım Kalite Güvencesi**

Yazılım kalite güvencesi bir test değildir. Yazılım kalite güvencesi, yazılım süreçlerinin uygun olduğuna dair güven sağlayan ve amaçları doğrultusunda yazılım ürünleri üreten yazılım süreçlerinin yeterliliğini değerlendiren faaliyetlerdir.

**2.2. Doğrulama ve Sağlama**

Doğrulama ve sağlamanın amacı, organizasyonun yaşam döngüsü boyunca sisteme kalite geliştirmesine yardımcı olmaktır. Doğrulama ve onaylama yaşam döngüsü boyunca ürün ve süreçlerin objektif bir değerlendirmesini sağlar. Ürünlerinin gereksinimlerine uygun olup olmadığını ve ürünün kullanım amacına, kullanıcı ihtiyaçlarına uygun olup olmadığını belirler.

**2.3. İncelemeler ve Denetimler**

İncelemeler ve denetim süreçleri, bu yapılar için kuruluş veya proje tarafından oluşturulan standartlara göre yazılım mühendisliği yapıtlarının incelenmesi anlamına gelir. Ürün güvencesi ve süreç güvencesi denetimleri genellikle Yazılımı Kalitesi ekiplerinden bağımsız yazılım kalite güvencesi personeli tarafından yürütülür.

**2.3.1. Yönetim İncelemeleri**

Yönetim incelemesinin amacı ilerlemeyi izlemek, plan durumunu belirlemek ve yönetim süreçlerinin ve tekniklerinin etkinliğini değerlendirmektir. Yönetim incelemeleri, projelerin veya bakım çalışmalarının durumunu belirleme planlarıyla gerçek proje sonuçlarını karşılaştırır. Yönetim değerlendirmelerinin ana parametreleri proje maliyeti, zamanlama, kapsam ve kalitedir.

**2.3.2. Teknik İncelemeler**

Teknik incelemenin amacı, bir yazılım ürününü, amaçlanan kullanıma uygunluğunu belirlemek ve standartlardaki tutarsızlıkları belirlemek için bir ekip tarafından değerlendirmektir. Projenin teknik durumunu doğrulamak için yönetime kanıt sağlar. Teknik incelemeler yazılım gereksinimleri ve yazılım tasarımı iş-ürünleri üzerinde gerçekleştirilir. Teknik inceleme, listelenen tüm faaliyetler tamamlandıktan sonra tamamlanır. Kaynak kodun teknik incelemeleri, algoritmaların analizi, kodlama standartlarına bağlılık, kodun yapısı gibi çok çeşitli konuları içerebilir.

**2.3.3. Denetimler**

Bir incelemenin amacı, yazılım ürün anomalilerini tespit etmek ve tanımlamaktır. Diğer teknik değerlendirme türlerine göre denetimlerin bazı önemli ayırıcıları şunlardır: Kurallar: Denetimler, kuruluş tarafından belirlenen kurallara göre bir iş ürününün incelenmesine dayanır. Farklı çalışma türleri için kurallar tanımlanabilir: gereksinim kuralları, mimari açıklamalar, kaynak kodu gibi. Örnekleme: Bir belgedeki her sözcüğü ve şekli incelemeye çalışmak yerine, denetleyicilerin incelenen belgelerin numuneleri değerlendirmesine olanak tanır. Akran: Teftiş ekibi üyeleri üzerinde yönetim pozisyonlarında bulunan kişiler denetime katılmaz. Moderatör: Teftiş teknikleri konusunda eğitilmiş tarafsız bir moderatör, teftiş toplantılarına öncülük eder. Toplantı: Teftiş süreci, teftiş ekibi üyelerinin buldukları anormallikleri ve diğer sorunları rapor ettikleri resmi bir prosedüre göre bir moderatör tarafından yürütülen toplantıları içerir. Denetimler genellikle ürünün nispeten küçük bir bölümünde aynı anda yapılır. Denetimler ayrıca bir denetim lideri, bir kaydedici, bir okuyucu ve birkaç müfettiş içerir.

**2.3.4. Geçişler**

Sistematik bir gözden geçirmenin amacı bir yazılım ürününü değerlendirmektir. Bir yazılım ürünüyle ilgili bir hedef kitleyi eğitmek amacıyla bir gözden geçirme yapılabilir.

**2.3.5. Süreç Güvencesi ve Ürün Güvence Denetimleri**

Yazılım denetiminin amacı, yazılım ürünlerinin ve süreçlerinin yönetmeliklere, standartlara, kurallara, planlara ve prosedürlere uygunluğunun bağımsız bir değerlendirmesini sağlamaktır. Süreç güvencesi denetimleri, proje hedeflerine ulaşmak için planların, zamanlamaların ve gereksinimlerin yeterliliğini belirler.

**3. Pratik Hususlar**

**3.1. Yazılım Kalite Gereksinimleri**

**3.1.1. Etki Faktörleri**

Çeşitli faktörler planlamayı, yönetimi ve tekniklerinin seçimini etkiler. Yazılımın bulunduğu sistemin etki alanı, sistem fonksiyonları, yazılım sisteminin bulunduğu fiziksel ortam, sistemde kullanılacak bileşenler, yazılım mühendisliği standartları, bütçe, personel, proje organizasyonu, planlar, sistemin bütünlük düzeyi gibi faktörler süreçlerinin nasıl organize edildiğini ve belgelendiği, belirli faaliyetlerinin nasıl seçildiğini, hangi kaynaklara ihtiyaç duyulduğunu etkiler.

**3.1.2. Güvenirlilik**

Sistem hatasının son derece ağır sonuçlar doğurabileceği durumlarda genel güvenilirlik üstündeki temel kalite gereksinimidir. Güvenilmez sistemler bilgi kaybına neden olabilir.

**3.1.3. Yazılımın Bütünlüğü**

Bütünlük düzeylerini tanımlamak risk yönetimi yöntemidir. Yazılım bütünlüğü düzeyleri, yazılım karmaşıklığını, kritikliğini, risk düzeyini, güvenlik düzeyini, istenen performansı, güvenilirliği gibi projeye özgü özellikleri temsil eden bir dizi değerdir. Aranan yazılım bütünlüğü düzeyleri, yazılım geliştikçe değişebilir. Sistemde veya yazılımda uygulanan tasarım, kodlama, yordam ve teknoloji özellikleri, aranan yazılım bütünlüğü düzeylerini yükseltebilir veya düşürebilir.

**3.2. Kusur Karakterizasyonu**

Yazılım kalite değerlendirme teknikleri, kusurları ve hataları bulabilir. Bu tekniklerin karakterizasyonu, ürünün anlaşılmasına sağlar, üründe düzeltmeleri kolaylaştırır ve yönetime ürünün durumu hakkında bilgi verir.

**3.3. Yazılım Kalite Yönetim Teknikleri**

İki kategori kullanır: statik ve dinamik.

**3.3.1. Statik Teknikler**

Statik teknikler, yazılım belgelerini (gereksinimler, ara yüz özellikleri, tasarımlar ve modeller dahil) ve yazılım kaynak kodunu kodu yürütmeden inceler. Kaynak kodu, denetim akışını analiz eden ve ölü kodu arayan araçlar, gibi araçlar statik çözümleme araçları olarak kabul edilir.

**3.3.2. Dinamik Teknikler**

Dinamik teknikler yazılım kodunun yürütülmesini içerir. Yazılımın geliştirilmesi ve bakımı boyunca farklı türde dinamik teknikler uygulanmaktadır. Genel olarak, bunlar test teknikleridir, ancak simülasyon ve model analizi gibi teknikler dinamik olarak kabul edilebilir.

**3.3.3. Test**

İki tür test Doğrulama ve Onaylama kapsamına girebilir: Projede kullanılacak araçların değerlendirilmesi ve testleri, üründe kullanılacak bileşenlerin ve ürünlerinin uygunluk testleri. Bazen bağımsız bir kuruluş test yapmak veya test sürecini izlemek için Doğrulama ve Onaylama tarafından testin kendisini değerlendirmek için görevlendirilebilir.

**3.4. Yazılım Kalite Ölçümü**

Yazılım kalitesi ölçümleri karar vermeyi desteklemek için kullanılır. Yazılımın artan gelişmişliğiyle, kalite soruları yazılımın ölçülebilir kalite hedeflerine ne kadar iyi ulaştığını değerlendirir. Yazılım kalitesi ölçümü ile desteklenen kararlar arasında yazılım kalitesi düzeylerinin belirlenmesi yer almaktadır. Genellikle, bir hatanın ne zaman bulunduğuna ve geliştirme sürecinde kusuru bulmaya göre kusuru düzeltmek için ne kadar çaba sarf ettiğine dayanan genel maliyet modelleri kullanılır. Yazılım kalitesi ölçümü, kusur oluşumlarının ölçülmesi ve en sık meydana gelen kusur türlerini anlamak için istatistiksel yöntemlerin uygulanmasını içerir. Bu bilgiler, yinelemelerini önlemek, azaltmak veya ortadan kaldırmak için yöntemleri belirlemek için yazılım süreci geliştirme tarafından kullanılabilir.

**4. Yazılım Kalite Araçları**

Yazılım kalite araçları statik ve dinamik analiz araçlarını içerir. Statik analiz araçları kaynak kodu girişi, kodu yürütmeden sözdizimsel ve anlamsal analiz yapmak ve kullanıcılara sonuç sunmaktır. Kaynak koduna ek olarak, modeller de dahil olmak üzere yapılara uygulanabilen statik analiz araçları çeşitlidir.