Yazılım Kalitesi ve Testi Stratejileri -1

18.05.2026

Yazılım Kalitesi ve Testi Stratejileri

- Yazılım testinin ürün geliştirmedeki stratejik öneminin açıklanması,
- Test faaliyetlerinin maliyet, kalite, zaman ve risk üzerindeki etkisinin tartışılması,
- Farklı test türlerini ve test seviyelerini ayırt edebilmesi,
- Agile, DevOps ve CI/CD süreçlerinde testin rolünün değerlendirilmesi

Yazılım Ürünü, Yazılım Kalitesi, Yazılım Testi ve İlişkileri

Yazılım ürünü; kullanıcıya, kuruma / pazara değer sunmak amacıyla geliştirilen, bakım yapılan, güncellenen ve yaşam döngüsü boyunca yönetilen dijital çözümdür.

Bir diğer ifade ile yazılım ürünü:

Fonksiyonel beklentileri olan,

Kullanıcı deneyiminin öne çıktığı,

Performans, güvenlik gibi fonksiyonel olmayan gereksinimlerin sağlanması gerektiği,

Bakımının yapılabildiği,

Ölçeklenebilirliğin sağlandığı

uzun bir geliştirme sürecidir.

Yazılım Ürünü, Yazılım Kalitesi, Yazılım Testi ve İlişkileri

Yazılım kalitesi farklı bakış açılarından değerlendirilir. Bunlar:

- Kullanıcı açısından kalite,
- Geliştirici açısından kalite,
- İş birimi açısından kalite,
- Operasyon ekibi açısından kalite

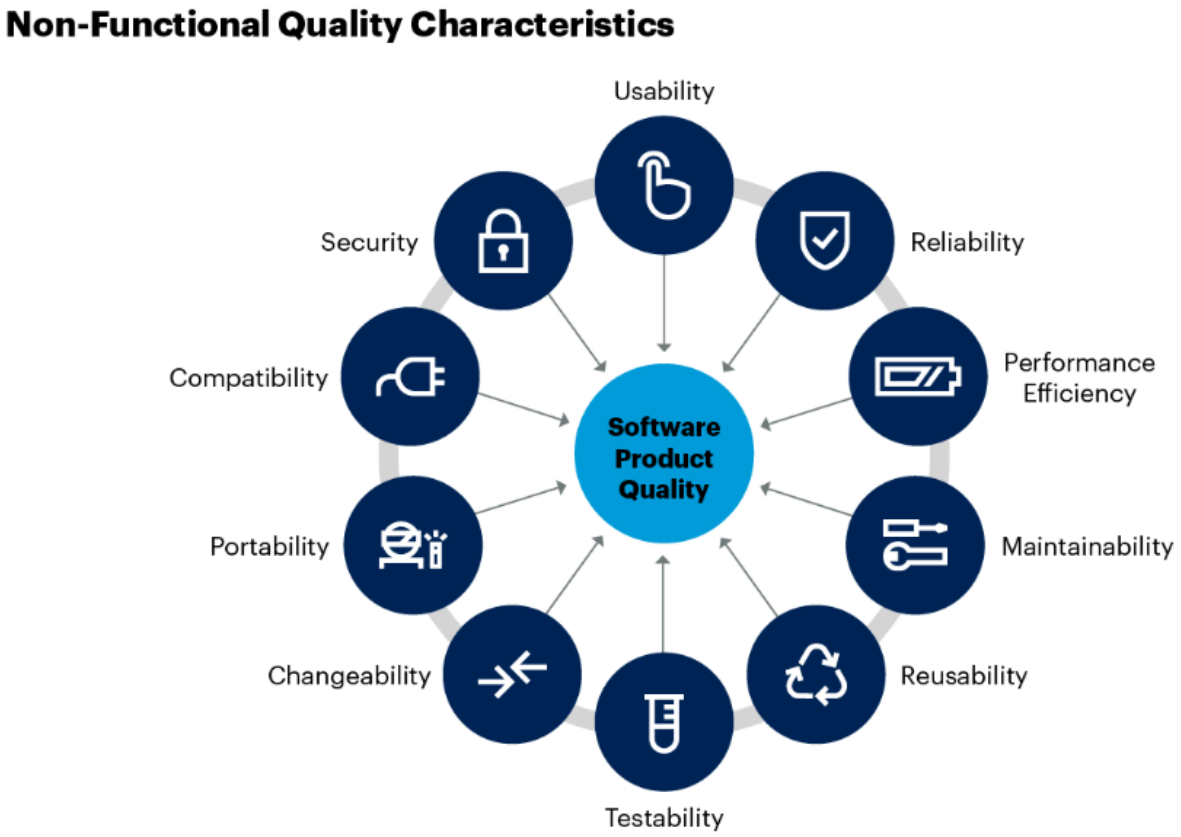
olarak farklı birimler için değerlendirilmelidir.

Yazılım kalitesi, yazılım ürününün belirlenen gereksinimleri, kullanıcı beklentilerini ve kalite özelliklerini ne ölçüde karşıladığı ile ilgilidir.

Bu yazılım, doğru işi doğru şekilde, güvenilir biçimde ve sürdürülebilir olarak yapıyor mu?

sorusunun cevabına göre değerlendirilir.

Fonksiyonel Olmayan Kalite Özellikleri



- Fonksiyonel olmayan testler, yazılım sisteminin kritik yönlerini doğrular.
- Bu tür testler aynı zamanda, yazılım ürünlerinin genel kalitesini ve güvenilirliğini artırmayı amaçlar.
- Teknolojik gelişmeler, değişen kullanıcı beklentileri, güvenilirlik, performans, güvenlik ve kullanılabilirlik taleplerindeki artış nedeniyle sürekli değişmektedir.

2022 Gartner Software Engineering survey, takım liderlerinin yarıdan fazlası kalite ve test becerilerinde eksiklik olduğunu kabul etmişlerdir.

Fonksiyonel Kalite Özellikleri

- Yazılımın beklenen işlevleri doğru şekilde yerine getirip getirmediğiyle ilgilidir.

Örneğin;

Kullanıcı sisteme giriş yapabiliyor mu?

Sepete eklenen ürün doğru fiyatla görünüyor mu?

Para transferi doğru hesaba yapılıyor mu?

Rapor ekranı doğru verileri gösteriyor mu?

Kampanya indirimi doğru hesaplanıyor mu?

- Fonksiyonel kalite, genellikle gereksinim dokümanları, kullanıcı hikâyeleri, iş kuralları ve kabul kriterleri üzerinden değerlendirilir.

Decentralize Quality Assurance Testing Teams to Drive Productivity

2024 - Gartner

- Modern yazılım kalitesi ölçütleri, geliştirmenin erken dönemlerinden itibaren ve daha sık test yapmayı gerektirir.
  - Merkezi test ekipleri, ürün geliştiren pek çok ekip için problem oluşturmaktadır.
- Takım liderleri, test uzmanlarını geliştirme ekiplerine dağıtılarak toplamda planlanmamış işlerden ve sürüm problemlerinden kurtulmayı hedefler.
- Dağıtılmış test uzmanları, ürün ekiplerine özerklik sağlar ve kendi kalitelerini belirleyebilir.
  - Ekibin ortak öğrenme sağlamaması olumsuzluk yaratır.
- Gartner Raporuna göre, şirketler içerisinde Kalite Güvence (QA- Quality Assurance) Mühendisi rolüne önem verilmeye başlamıştır.

Platform Mühendisliği

- Yazılım ürünü geliştirilirken popüler bir yaklaşım haline gelmiştir.
  - Ama, test süreci bu yaklaşıma dahil edilememiştir.
  - Bu da yazılım kalitesi **uygulamalarının** ve **araçlarının** ölçeklenebilirliğini düşürmektedir.
- Dağıtılmış test uzmanları, ürün ekiplerine özerklik ve kendi kalitelerini belirleme olanağı sağlamaktadır.
  - Ancak bu durumda test uzmanları ortak öğrenmeyi gerçekleştiremez.

Backstage

- Spotify tarafından geliştirilen açık kaynaklı bir platform mühendisliği aracıdır.
  - Yazılım ekipleri tarafından merkezi Internal Developer Portals IDP) oluşturmada açık kaynaklı bir araçtır.
- Geliştiricilerin, uygulamalarını oluşturmada ve yönetmede gerekli tüm araç (tool) ve servislere erişmelerini sağlayan birleşik bir arayüzdür.
  - Diğer araçlarla entegrasyonu kolaydır.
- Backstage, ekipler arasında iş birliğini kolaylaştırarak görünürlüğü ve bilgi paylaşımını iyileştirir.

Yazılım Ürünü, Yazılım Kalitesi, Yazılım Testi ve İlişkileri

- Yazılım testinin temel amacı:
  - Hataların bulunması,
  - Risklerin görünür kılınması,
  - Ürüne olan güvenilirliğin oluşturulması,
  - Canlıya alma kararının desteklemesidir.

- Yazılım testinin yapılması, hatasız yazılım garantisi anlamına gelmez.
- Geliştirilen yazılım ürünün risklerinin belirlenmesiyle daha gerçekçi kararlar alınabilir.

Yazılım Hatalarının Tipleri

- Yazılım hatalarının maliyeti farklı değerlendirilir. Bunlar:
  - Analiz aşamasında bulunan hatalar,
  - Geliştirme aşamasında bulunan hatalar,
  - Test aşamasında bulunan hatalar

olarak çeşitlenmiştir.

- Canlı ortamdaki yazılım hatası ise, kullanıcı tarafından bulunan hatalar şeklinde karşılaşılır.

Yazılım Kalitesine ilişkin Temel Terimler

- **Error** (Hata/Yanlışlık) ile ilgili en yaygın örnek kodun yazılması sırasında yapılan yanlışlıktır.
  - Genel ifadesiyle geliştirici, analist, tasarımcı, test uzmanı veya kullanıcı bir tanımlanmış bir gerçeği hatalı anlayarak yanlış yaptığında **error** oluşur.
- **Fault** (Bozukluk / Kusur): İlgili adımda kişinin sistem içerisinde yanlış bir çözüm yapması, çözümün bozuk / kusurlu olarak gerçekleşmesidir.
  - Tasarım aşamasında, verinin modellenmesinde veya dokümanın oluşturulmasına çözüm doğru gerçekleştirilmediğinde **fault** oluşacaktır.
- **Defect,** Yazılım ürününün gereksinimlere, tasarıma, standartlara veya beklenen davranışa uymamasıdır.
  - Test süreçlerinde sık kullanılan terimdir.
  - Test uzmanı bir problem bulduğunda bunu **defect** olarak raporlar.
    - ✓Genel olarak ürünün beklenen davranıştan veya gereksinimden sapmasıdır.

Yazılım Kalitesine ilişkin Temel Kavramlar

- **Bug,** yazılım hatası için yaygın olarak kullanılan ve yazılım ekibi arasında kullanılan bir terimdir.
  - Teknik olarak *defect* veya *fault i*le benzer anlamda kullanılır.
- Kurumsal test süreçlerinde **defect** yaygın ifadedir; test yönetimi süreçlerinde kullanılır.
- Geliştirme ekibi "bug açtım" der; test ekibi "defect kaydı oluşturdum" der.
- **Failure**, yazılım çalıştırıldığında sistemin beklenen davranışı göstermemesi, yani dışarıdan gözlemlenenbaşarısızlıktır (sistemin çökmesi gibi).
  - Yazılımın içindeki fault veya defect, kullanıcı ya da testçi tarafından sistem çalışırken fark edildiğinde sonuç **failure** olur.
- **Detect,** Yazılım testinde bir hata, kusur veya başarısızlığı bulma eylemidir.

Terimlere ilişkin bir Örnek

Error: Geliştiricinin “500 TL ve üzeri” kuralını yanlış yorumlaması

Fault : Kodda > kullanılması gerekirken >= mantığının uygulanmaması

Defect: 500 TL sepet tutarında kargonun ücretsiz olmaması

Bug: Ekip içinde “ücretsiz kargo bug’ı” diye adlandırılan sorun

Failure: Kullanıcının 500 TL alışverişte kargo ücreti görmesi

Detect: Testçinin 500 TL sınır değerinde hatayı tespit etmesi

Mobil Bankacılık Örneği

Error: Yetkilendirme kuralının yanlış analiz edilmesi veya yanlış kodlanması,

Fault: Kodda kullanıcıya ilişkin ID kontrolünün eksik yapılması,

Defect: Yetkisiz veri görüntüleme kusuru,

Bug: "Kullanıcı başka hesapları görebiliyor" hatası,

Failure: Canlı ortamda kullanıcının başka kişinin hesap bilgisini görmesi,

failure yalnızca teknik bir başarısızlık değil; aynı zamanda güvenlik, mahremiyet, regülasyon ve itibar riskidir.

Detect: Güvenlik testi veya kullanıcı bildirimiyle durumun tespit edilmesi.

Kalitenin Sağlanmasında Paydaşların Önemi

- **Son kullanıcının** sistemi kolay, hızlı ve sorunsuz kullanımıdır. "İşimi rahatça yapabiliyor muyum?" sorusunun cevabı beklenir.
- **Müşteri / iş biriminin** iş ihtiyacının karşılanma ölçüsüdür. "Bu ürün iş hedefime hizmet ediyor mu?" sorusuna cevap beklenir.
- **Geliştirici** için temiz, anlaşılır ve sürdürülebilir kod yazılmasıdır. "Bu sistemi geliştirmek kolay oldu mu?" sorusuna cevap beklenir.
- **Test uzmanı** için, gereksinimlere uygunluk ve risklerin görünürlüğüdür. "Hangi riskler test edildi?" sorusuna tam cevap beklenir.
- **Operasyon ekibi** için, kararlılık, izlenebilirlik ve kesintisiz çalışmadır. "Sistem canlıda sorunsuz çalışıyor mu?" sorusuna cevap beklenir.
- **Yönetim** için, maliyet, zaman, itibar ve müşteri memnuniyetidir. "Bu ürün kuruma değer sağlıyor mu?" sorusuna cevap beklenir.
- **Güvenlik ekibi** için, veri ve işlem güvenliğidir. "Sistem saldırılara karşı dayanıklı mı?" sorusuna cevap beklenir.

Yazılım Testi: Yaptıkları & Yapmadıkları

- Hataları ortaya çıkarmasına rağmen, hatasızlığı garanti edemez.
- Riskler hakkında bilgi vermesine rağmen, tüm olası risk durumlarını kapsayamaz.
- Kalite hakkında bilgi vermesine rağmen, kaliteyi tek başına yaratamaz.
- Karar vermeyi desteklemesine rağmen, iş hedeflerini tek başına belirleyemez.
- Geliştirme sürecini iyileştirmesine rağmen, kötü gereksinimleri düzeltemez.

Test, kalitenin kanıtı değil; kalite hakkında bilgi üreten bir faaliyettir.

Bir ürün testlerden geçmiş olabilir, ancak bu onun tüm koşullarda hatasız çalışacağı anlamına gelmez.

Test, belirli koşullar altında ürünün davranışını gözlemler.

Kaliteyi Etkileyen Faktörler

- Yazılım kalitesi, yalnızca test aşamasında belirlenmez. Kaliteyi etkileyen birçok faktör vardır.
  - Gereksinimlerin açıklığı, yanlış veya eksik geliştirmeyi azaltarak etkiler.
  - Mimari tasarım, performans, güvenlik ve bakım kolaylığını etkileyerek kaliteye katkı sunar.
  - Kod kalitesi, hata oranını ve bakım maliyetini etkiler
  - Test yaklaşımı, risklerin anlaşılabilirliğini artırır.
  - Ekip iletişimi, yanlış anlamaları azaltır
  - Kullanıcı geri bildirimi, ürünün gerçek ihtiyaca uyumunu artırır
  - Operasyon ve izleme, canlı ortam kalitesini sağlar.
  - Süreç olgunluğu, tekrarlanabilir ve güvenilir çıktılar üretir

Sonuç olarak:

- Kalite, yazılım geliştirme sürecinin sonunda kontrol edilen bir özellik değildir.
- Ürün geliştirmenin başından itibaren tasarlanan ve süreç boyunca aktif olan bir süreçtir.

Testin Erken Başlaması

Gereksinimlerin gözden geçirilmesi,

Kullanıcı hikâyelerinin netleştirilmesi,

Kabul kriterlerinin yazılması,

Risklerin belirlenmesi,

Test senaryolarının tasarlanması,

Test verilerinin planlanması için test erken başlar.

Niçin?

- ❑ Belirsiz gereksinimler erken fark edilir.
- ❑ Yanlış «business rules» geliştirilmeden düzeltilir.
- ❑ Test edilebilir gereksinimler oluşturulur.
- ❑ Geliştirici, analist ve test uzmanı ortak davranır.
- ❑ Geç aşamada yapılan maliyetli değişiklikler azalır.

Yazılım Testi ve Kalitesi: Önemli Yanlışlar

- Test ekibi varsa kalite sorunu olmaz.

Test ekibi hataları görünür kılar; ancak kalite tüm ekibin sorumluluğudur. Kötü gereksinimler, zayıf mimari veya aceleyle yazılmış kod yalnızca test ekibiyle telafi edilemez.

- Test en son yapılır.

Test, gereksinimlerin tartışıldığı ilk aşamalardan itibaren başlamalıdır. Geç başlayan test, çoğu zaman geç bulunan hata anlamına gelir.

- Tüm hataları bulmak mümkündür.

Gerçek sistemlerde tüm olası durumları test etmek genellikle imkânsızdır. Bu nedenle risk bazlı yaklaşım gerekir.

- Otomasyon varsa manuel teste gerek yoktur.

Otomasyon tekrar eden ve belirli testler de önemlidir; ancak keşifsel (Exploratory) test , yazılım geliştirme sürecinde test tasarımı ve yürütmenin eş zamanlı olarak yapıldığı dinamik yaklaşımda insan değerlendirmesi hala önemlidir.

- Çalışıyorsa kalitelidir. Bir yazılımın çalışması, kaliteli olduğu anlamına gelmez. Yavaş, anlaşılması zor veya bakımı imkânsız bir sistem de çalışıyor olabilir.

Online Bilet Satış Sistemi: Canlı Ortam Hatası

Bir online bilet satış platformu, yoğun talep gören bir konser için satışa çıkmıştır. Satış başladıktan kısa süre sonra bazı kullanıcılar ödeme yaptıkları halde biletlerinin oluşmadığını fark etmiştir. Bazı kullanıcılar ise aynı koltuğun birden fazla kişiye satıldığını bildirmiştir.

Bu problem yalnızca yazılım hatası mıdır, yoksa ürün kalitesi sorunu mudur?

Hangi kalite özellikleri zarar görmüştür?

Hangi test türleri bu riski azaltabilirdi?

Canlı ortamda bu hatanın kuruma etkileri nelerdir?

Bu olaydan sonra ürün geliştirme sürecinde ne değiştirilmelidir?

Online Bilet Satış Sisteminde Canlı Ortam Hatası

Bu vakada etkilenen **kalite özellikleri** şunlardır:

Fonksiyonel doğruluk

Güvenilirlik

Performans

Veri tutarlılığı

Kullanıcı güveni

Marka itibarı

- Bu hata yalnızca ödeme sonrası bilet oluşmaması şeklinde teknik olarak düşünülemez.
- Kullanıcı deneyimi, finansal güven, operasyonel yük ve marka değeri de etkilenmiştir.

Yazılım Test Araçları ve Test Otomasyonu

https://coderspace.io/blog/yazilim-test-araclari-ve-test-otomasyonu/