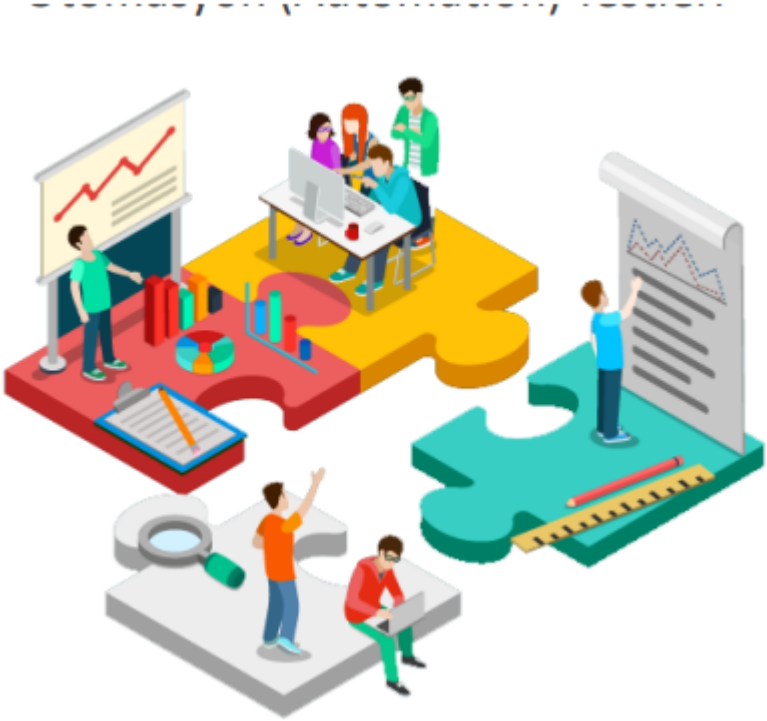
Sınama Test Stratejileri Nedir



(https://i2.wp.com/oktayolgun.com.tr/wp-content/uploads/2020/05/00GXr8.png?fit=736%2C372&ssl=1)

by **oktayolg (https://oktayolgun.com.tr/author/oktayolg/)**Mayıs 11, 2020

Sınama Test Stratejileri Nedir .

Test Stratejileri dört grup ile adlandırılmaktadır.

* Yazılım Testi.
* Test Tipleri.
* Test Metodları.
* Test Seviyeleri.

1. Yazılım Testi

Yazılımları test etmek için kullanılabilecek birçok yöntem vardır. Örneğin, sistem tamamen inşa edilene kadar beklenilir ve hataları bulma amacıyla tüm sistem üzerinde testler yapılır. Bu yaklaşım ilgi çekici olmasına rağmen çalışmaz. Tüm taraflarda memnuniyetsizlik uyandıracak hatalı bir yazılımla sonuçlanır. Diğer bir örnek ise, test günlük olarak, sistemin herhangi bir parçası inşa edildiğinde gerçekleştirilir. Bu yaklaşım daha az ilgi çekici olmasına rağmen çok verimlidir. Ne yazık ki birçok yazılım geliştirici, bu yöntemi kullanmakta endişe duymaktadır.

Yazılım Testinin Yapılma Nedenleri

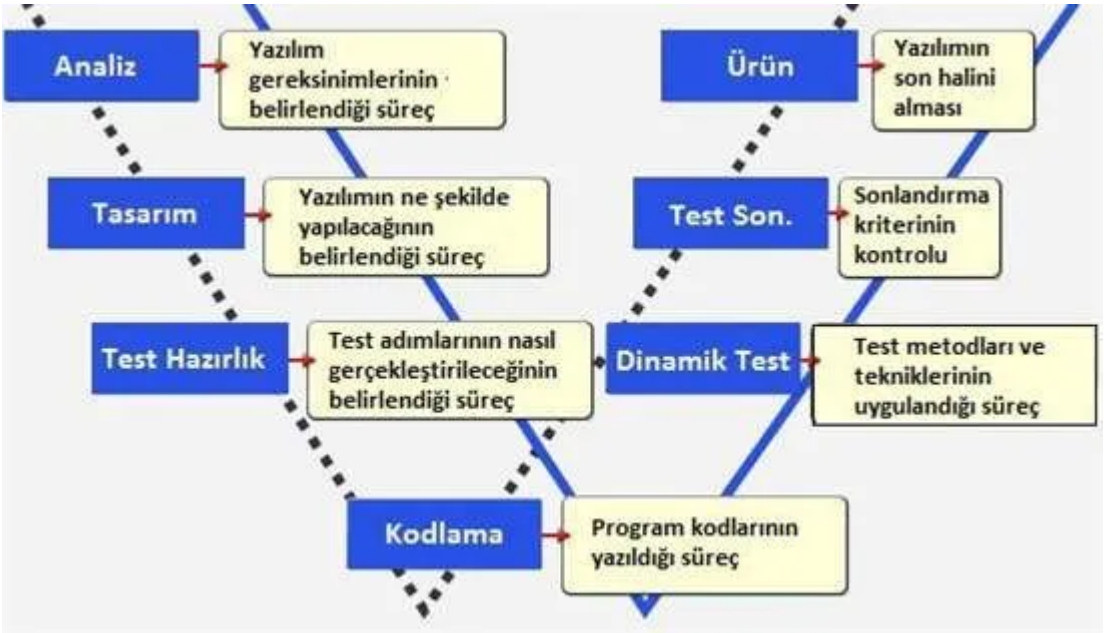
Geliştirilen yazılımlarda, kaynağı ve sebebi değişmekle birlikte çeşitli hataların olması kaçınılmazdır. Çalışılan uygulama alanının kritikliğine göre bu hataların doğuracağı sonuçların biçimi değişebilmektedir. Bir finans uygulamasında yapılan küçük bir hata çok büyük miktarlarda para kaybına yol açabilecekken, bir askeri uygulamada yapılması muhtemel küçük bir hata mal kaybının yanı sıra can kaybına da neden olma riskini taşımaktadır. Bununla birlikte uygulamanın türüne bağlı olarak yazılım testleri,

* Müşteriye sunulmadan önce ürün kalitesinden emin olmak,
* Yeniden çalışma (düzeltme) ve geliştirme masraflarını azaltmak,
* Geliştirme işleminin erken aşamalarında yanlışları saptayarak ileri aşamalara yayılmasını önlemek, böylece zaman ve maliyetten tasarruf sağlamak,
* Müşteri memnuniyetini arttırmak ve izleyen siparişler için zemin hazırlamak,

amaçları doğrultusunda da yapılmaktadır. Ayrıca ürettikleri yazılımları yeterince test etmeden müşterilerinin hizmetine sunan firmalar, olası yazılım hataları sonucunda itibar kaybedecekleri gibi müşterilerine tazminat ödemek durumunda da kalabilirler.

Yazılım Test Süreci

Genel olarak yazılım projeleri analiz – gt; tasarım – gt; kodlama – gt; test – gt; ürün süreçleri izlenerek geliştirilir. Bütün süreçler birbirini bu şekilde izlese de test süreci hiçbir zaman kodlama sürecinin bitmesini beklemez. İdeal bir yazılım test süreci, analiz – gt; tasarım – gt; **test hazırlık süreci**– gt; kodlama – gt; **dinamik test süreci** – gt; **testin sonlandırılması** – gt; ürün şeklinde olmak durumundadır. Test süreçlerince yapılması gereken işlemler şu şekilde sıralanır:



2. Test Tipleri

Yazılım geliştirme yaşam döngüsü boyunca kullanılan test tipleri:

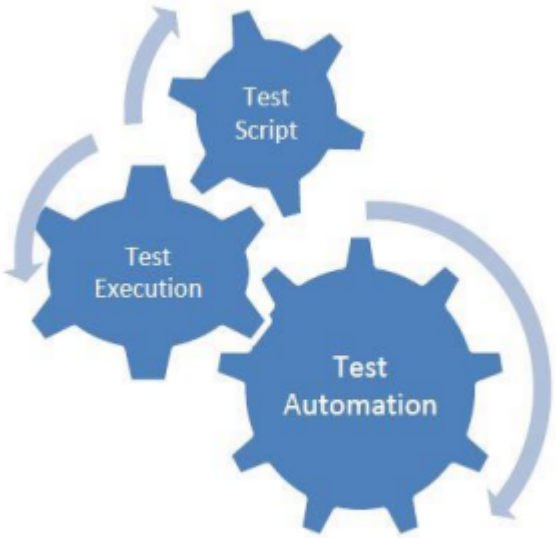
* El ile yapılan (Manual) Testler.
* Otomasyon (Automation) Testleri

Manuel Test

Yazılımın herhangi bir test otomasyonu veya herhangi bir script kullanılmadan el ile ile test edilmesidir. Yazılımdaki herhangi bir beklenilmeyen davranışı veya hatayı bulmak için yapılır. Küçük projelerde kullanılması daha uygundur.

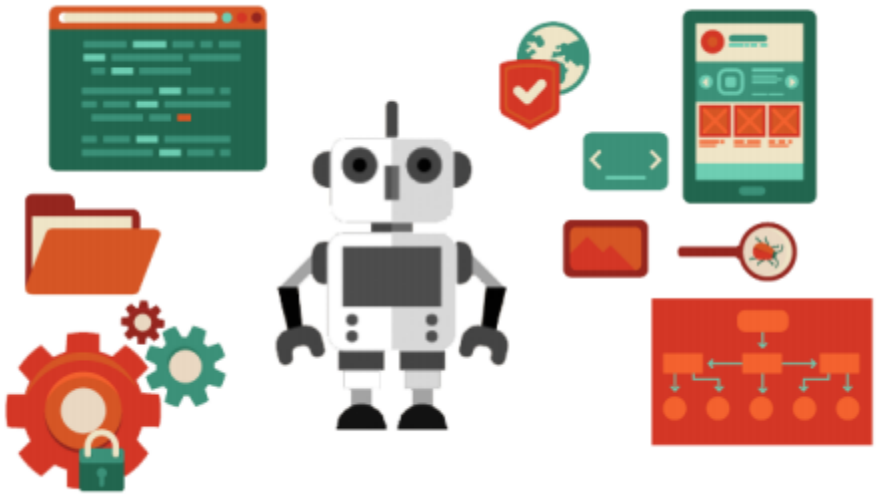
Otomasyon Testi

Test otomasyonu olarak da bilinen otomasyon testi, test yapan kişilerin scriptleri yazdığı ve yazılımı test etmek için başka yazılımlar kullandığı test türüdür. Otomasyon testi, test senaryolarının hızlı, art arda ve tekrarlarca uygulanabilmesini sağlar. Yük, performans, stress gibi çok kullanıcı gerektiren testlerde ve sık sık değişiklik yapılan regresyon testlerinde kolaylık sağlar.



Test Hazırlık Süreci

Bu süreç, yazılım test süreçleri içindeki ilk aşama olmakla beraber, testin efektif sonuçlar vermesi ve verimli olması açısından büyük öneme sahiptir. Dolayısıyla, bir yazılımı iyi test edebilmek için, test işlemlerinden önce, sağlıklı bir test hazırlık süreci kaçınılmazdır. Test hazırlık sürecinde yapılması gereken birtakım standart işlemler şu şekilde sıralanır:

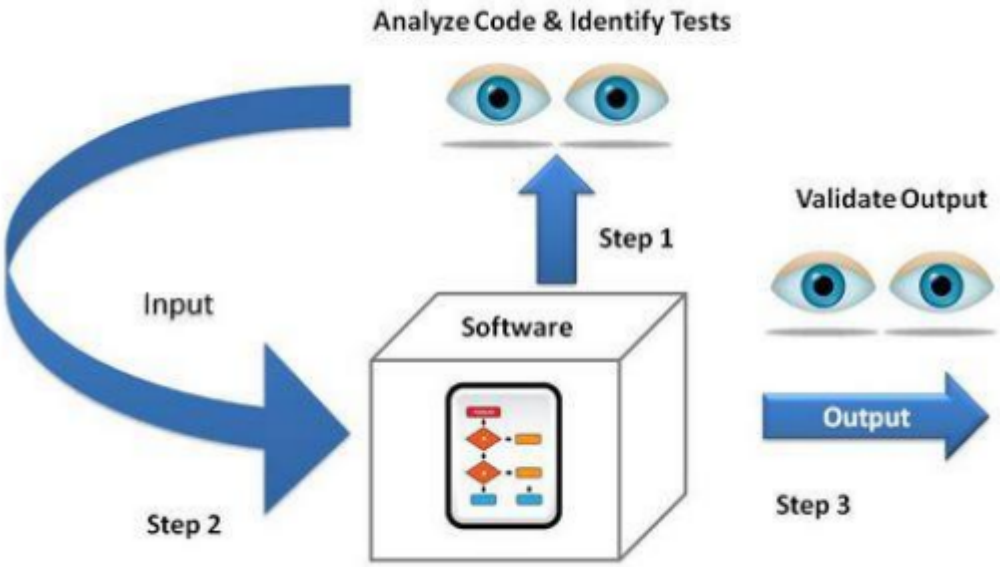


* Öncelikle test edilecek yazılıma ait analiz ve teknik tasarım aşamaları ile ilgili dökümanlar, test ekibi tarafından incelenir.
* Yazılım içinde test edilecek ve edilmeyecek modüller belirlenir.
* Risk analizi yapılır ve yapılan değerlendirmeye göre dinamik test aşamasında uygulanacak olan test teknikleri ve metodları belirlenir.
* Dinamik testin uygulanacağı ortamlar ve bu ortamların ihtiyaçları belirlenip, uygun şartlar sağlanır.
* Test ekibi içinde görev paylaşımı ve zaman planlaması yapılır.
* Testin sonlandırma kriterleri belirlenir.
* Bir programa belirli girdiler (input) verildiğinde hangi çıkışların (output) ne şekilde alınması gerektiğini bildiren test case senoryaları belirlenir.
* Dinamik testin hangi adımlarla ve ne şekilde uygulanacağının belirtildiği test planı hazırlanır.

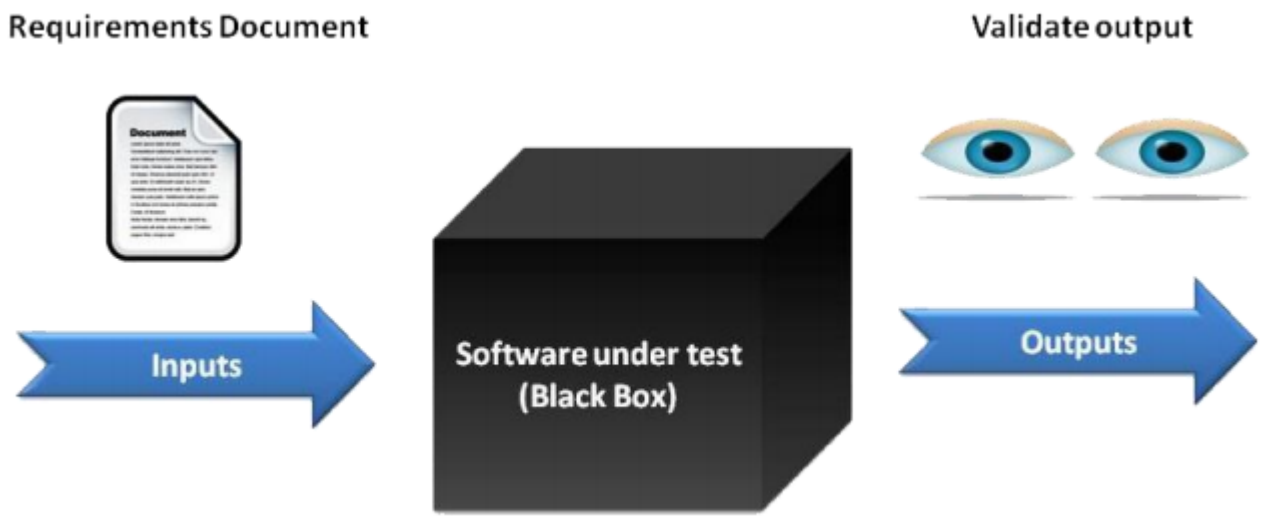
Yukarıda sıralanan test hazırlık sürecine ait aşamalar gerçeklendikten sonra dinamik yazılım testi aşamalarına geçilir.

Gri Kutu (Gray Box Testing Technic) Kara kutu ve beyaz kutu testlerinin birleşimidir. Test uzmanının veri tabanına ve dokümanlara erişimi vardır. Böylece tasarıma ve verilere uygun test dokümanı üretebilir. Yani uygulamanın iç işlemlerine kısmen erişime izin verir.

Beyaz Kutu Test Tekniği (White Box Testing Technic) : Beyaz kutu test tekniğinin en genel tabiri kod testidir. Projenin hem kaynak kodu, hem de derlenmiş kodu test edilir. Bu tür testler, uygulama kodunun iç mantığı üzerindeki bilgiye bağlıdır. Yazılım kodundaki deyimler, akış denetimleri, koşullar vb. elemanlar sınanır.



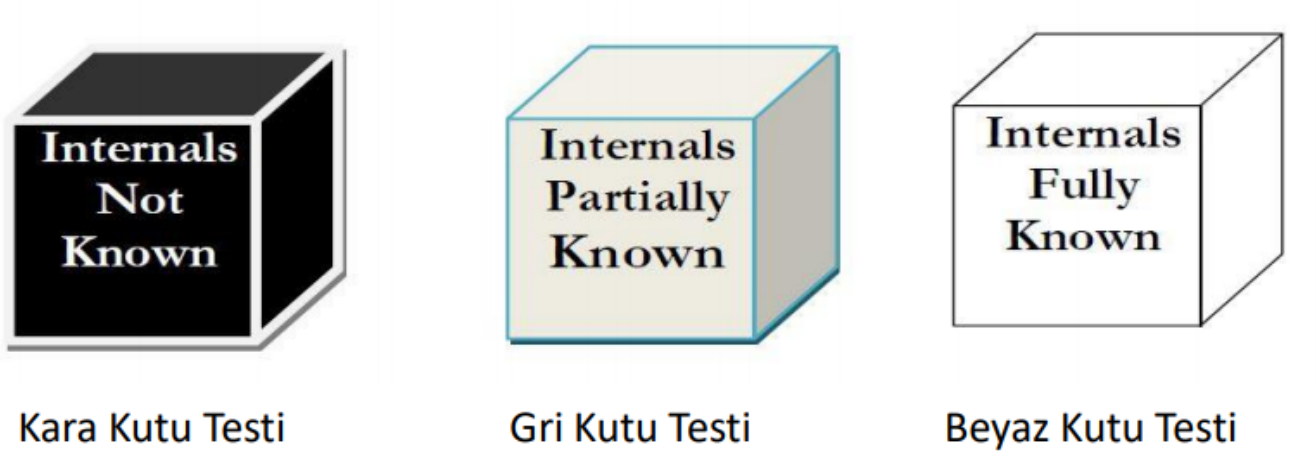
Kara Kutu Test Tekniği (Black Box Testing Technic) : Test ekipleri tarafından en çok kullanılan teknik olan kara kutu test tekniği adından da anlaşılacağı gibi uygulamanın sadece derlenmiş kodu üzerinden test edilmesi olarak bilinir. Bu test tekniğinde, yazılımın programatik yapısı, tasarımı veya kodlama tekniği hakkında herhangi bir bilgi olması gerekli değildir. Yazılımın gereksinimine duyulan şeylere yanıt verip veremediği ve işlevselliği sınanmaktadır.



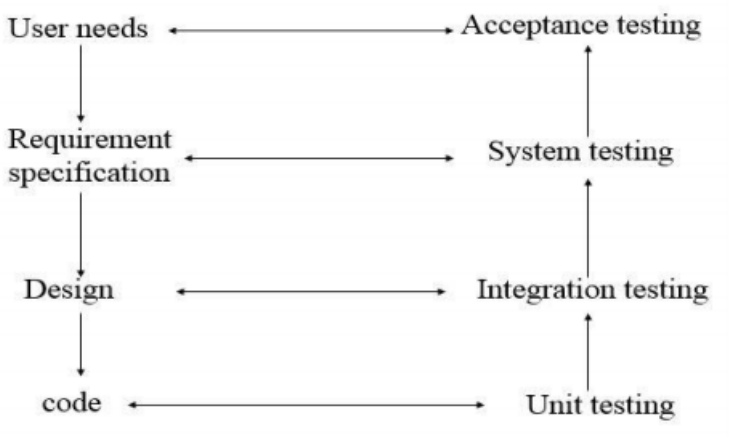
Testin Sonlandırılması

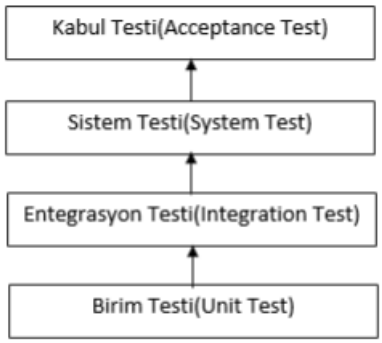
Yapılan testler sonucunda bulunan hatalar düzeltildikten sonra test sonlandırma kriterleri (test hazırlık süreci) kontrol edilir. Eğer tüm kriterlerin kabul edilebilir düzeyde sağlandığı tespit edilirse test sonlandırılır. Testin sonlandırılmasının ardından uygulama müşteri testine açılır (Kullanıcı Kabul Testi). Müşterilerin bulduğu hatalar veya değiştirilmesi istenilen noktalar gözden geçirilerek tekrar test ekibinin kontrolüne sunulur. Bu kontrolden çıkan uygulama ürün aşamasına geçer ve böylelikle yazılım test süreci sona erdirilerek, yazılım geliştirme sürecinin son basamağına geçilmiş olunur.

Metorların Karşılastırılması



Test Seviyeleri

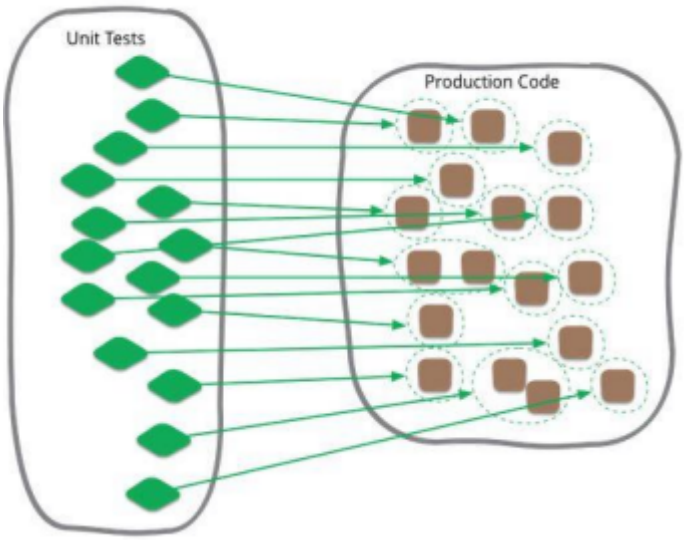




Yazılım Test Seviyeleri(Software Test Levels)

Birim(Unit) Testi

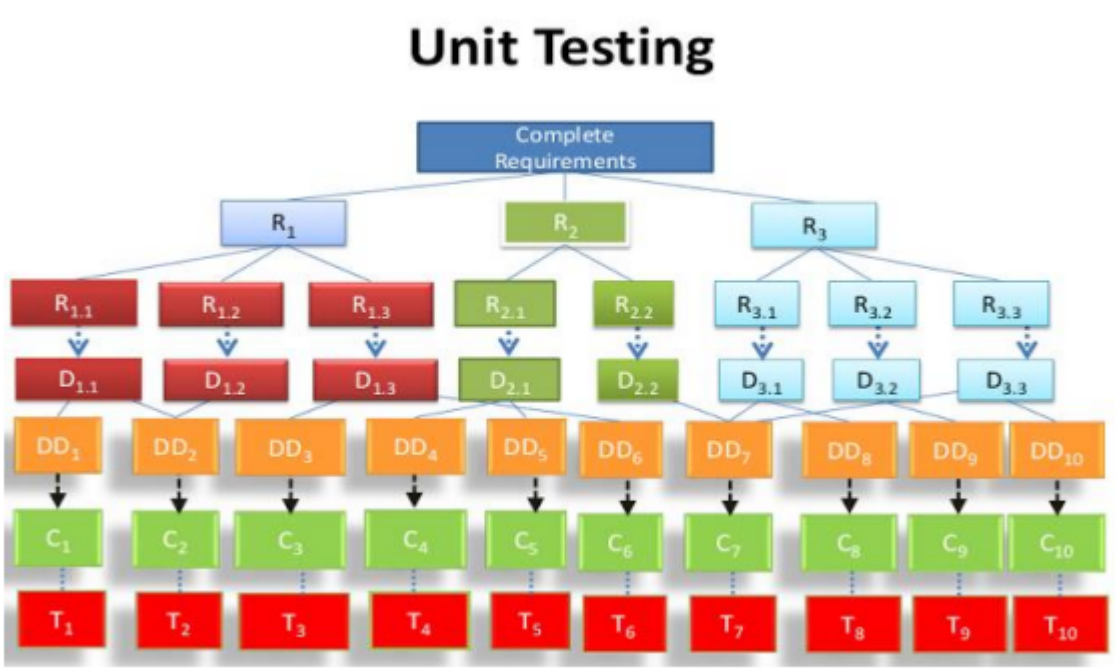
Birim testi, yazılım tasarımının en küçük biriminin (yazılım bileşeni yada modül) doğrulanmasıdır. Önemli kontrol yolları, modülün sınırları içerisindeki hataları ortaya çıkarmak için test edilir. Birim testi bir bileşenin sınırları içindeki mantık ve veri yapıları gibi iç işlemler üzerinde çalışır. Bu test türü birden fazla bileşen için paralel olarak gerçekleştirilebilir.



Birim Testi (Unit Testing) : Dinamik test sürecinin ilk aşaması olmakla beraber, hataların erken bulunup düzeltilebilmesi açısından da bu sürecin en önemli aşamasını oluşturur. Mikro ölçekte yapılan bu testte, özel fonksiyonlar veya kod modülleri (fonksiyonlar, veri yapıları, nesneler vb.) test edilir. Bu test, test uzmanlarınca değil programcılar tarafından yapılır ve program kodunun ayrıntıları ile içsel tasarım biçiminin bilinmesi gerekir. Uygulama kodu çok iyi tasarlanmış bir mimaride değilse oldukça zor bir testtir.

Birim test durumları

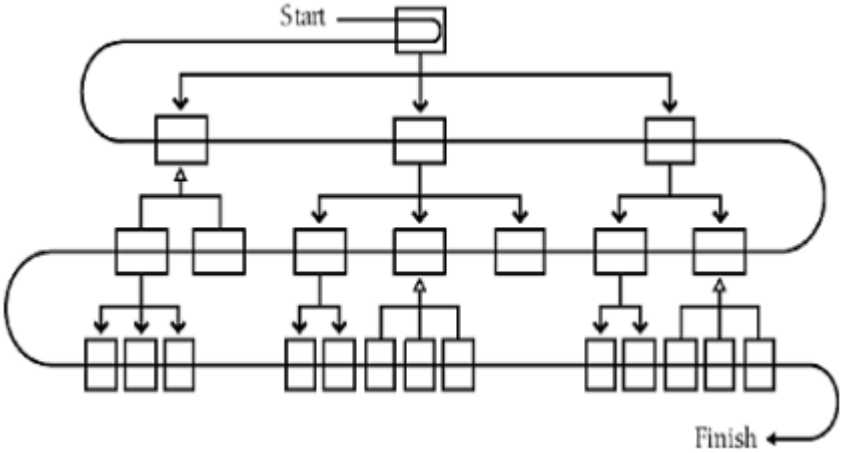
* Ara yüz
* Yerel veri yapıları
* Sınır koşulları
* Bağımsız yollar
* Hata yakalama yolları



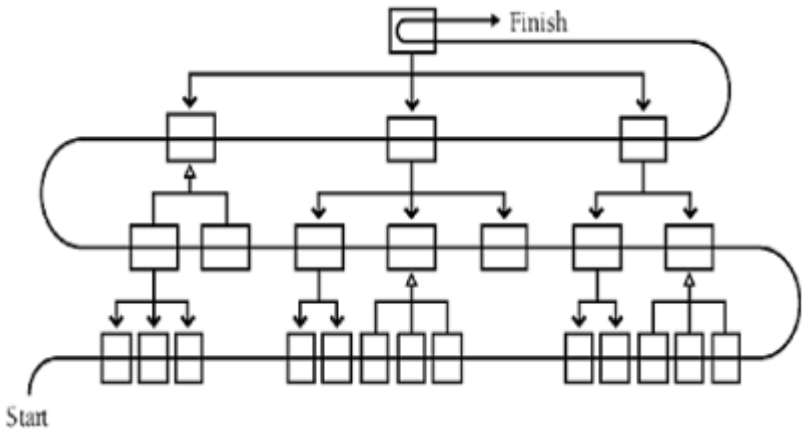
– Birim ara yüzü test edilerek bilgi giriş/çıkışlarının uygun ve yeterli şekilde yapıldığı kontrol edilir. Örneğin, programa giren ve çıkan tüm iletilerin doğru tipte oldukları gerçeklenir.
– Yerel veri yapıları incelenerek algoritmanın çalışması boyunca ya da yordamların çağrılması sırasında verilerin saklandığı yerin bütünlüğünün bozulup bozulmadığı test edilmelidir.
– Sınır koşullarının en düşük ve en yüksek değerleri, bu değerlerin biraz altı ve biraz üstü kullanılarak sınanmalıdır.
– Birim içindeki birbirinden bağımsız tüm çalışma yolları, tüm dallanmalar tek tek sınanmalıdır.
– Birim içindeki tüm hata yakalayıcılar birer birer denenmelidir.
Daha çok beyaz kutu yönteminin kullanıldığı birim testi ile düzgün ve hatasız çalıştığına kanaat getirilen bir birim artık tümleştirme testi için hazır hale gelmiş olur.
* Yazılan kodun her satırının başka bir kod tarafından test edilmesini sağlar.
* Kodun anlaşılmasını kolaylaştırır. Daha hızlı yazılım geliştirmeyi sağlar.
* Koddaki hata oranını azaltır. Kodların kalitesinin artmasını sağlar.
* Hataların çabuk tespit edilip düzenlenmesini sağlar.

Tümleyim Testi (Integration Testing) :
Bir uygulamanın farklı bileşenlerinin beraberce uyum içinde çalışıp çalışmadığını sınamak için yapılan bir testtir. Bileşenler, modüller, bağımsız uygulamalar, istemci/sunucu uygulamaları biçiminde olabilirler. Bu tür testlere, özellikle istemci/sunucu uygulamaları ve dağıtık sistemlerin testinde başvurulmaktadır. Bunun yanısıra uygulamaya yeni işlevsel elemanlar ya da program modülleri eklendikçe sürekli test edilmesi işlemine de "Artımsal Tümleyim Testi" adı verilir. Test uzmanları ve/veya programcılar tarafından gerçekleştirilen testlerdir.
Birimler bir anda tümleştirmek yerine artırımlı olarak tümleştirmek daha iyi sonuç verir. Artırımlı tümleştirme yönteminde değişik stratejiler kullanılabilir. Bunlar
* Yukarıdan aşağı tümleştirme
* Aşağıdan yukarıya tümleştirme
* Geri çekilme(Regression) testi

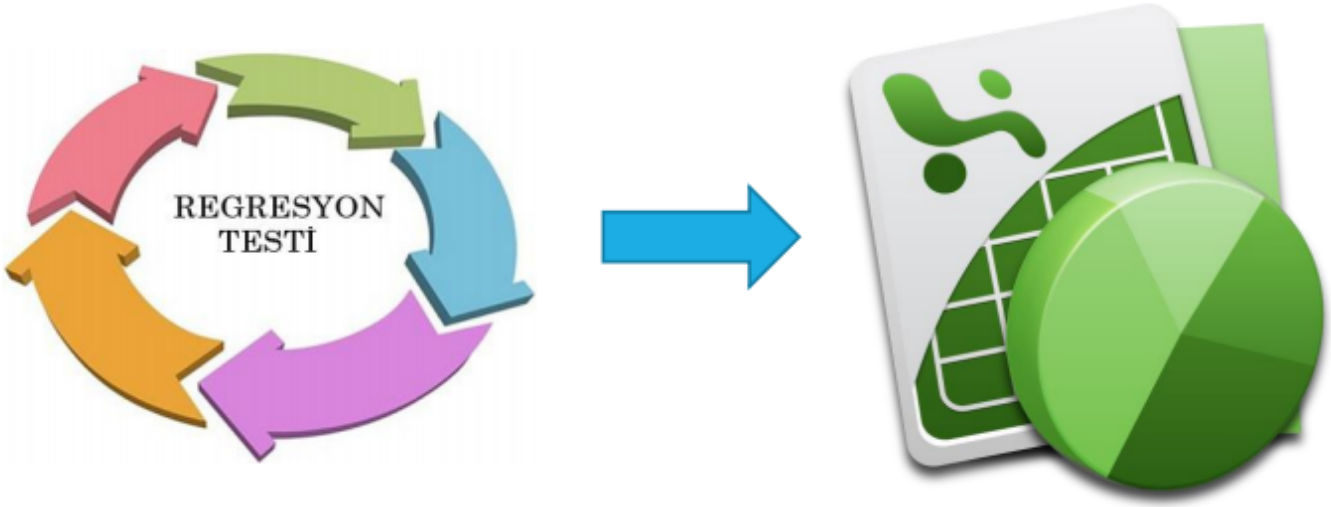
Yukarıdan Aşağı Tümleştirme (Top-Down Integration Testing) :
Bu stratejide, önce ana denetim biriminin testi yapılır, sonra ona en yakın düzeydeki birimlerden biri ile beraber test yapılır.



Aşağıdan Yukarıya Tümleştirme (Bottom-Up Integration Testing):
Alt düzey birimler birleştirilerek kümeler haline getirilir. Bu kümeler test edilir. Daha sonra bu kümelerin birleştirilmesinden oluşan daha üstü düzeyde kümeler meydana getirilir. Bu şekilde en üstte bulunan ana birime kadar ulaşılır.



Regresyon Testi (Regression Testing) :
Uygulamada ve uygulama ortamlarında gerekli değişiklikler ve sabitlemeler yapıldıktan sonra yeniden yapılan testlere çekilme (regresyon) testi denilir. Böylece, önceki testlerde belirlenen sorunların giderildiğinden ve yeni hatalar oluşmadığından emin olunur. Uygulamanın kaç kez yeniden test edilmesi gerektiğini belirlemek güçtür ve bu nedenle, özellikle uygulama geliştirme döneminin sonlarına doğru yapılır.



Sistem Testi

Aşağıdaki test türlerini içerir.

* Performans Testi Yük Testi
* Germe Testi
* Kurtarma Testi
* Taşınabilirlik Testi
* Kullanılabilirlik Testi

Zorlanım – Performans Testi (Performance Testing):

Bu test, çoğu kez "yük testi" ile aynı anlamda kullanılmaktadır. Aynı zamanda, beklenmedik (normal olmayan) ağır yükler, belirli eylemler ve taleplerin çok fazla artışı, çok yoğun sayısal işlemler, çok karmaşık sorgulamalar vb. ağır koşullar altında olan bir sistemin işlevsellik testi (iş yapabilme testi) olarak da tanımlanabilmektedir. Bir web sitesi için sistem tepkisinin hangi noktada azaldığı veya yanıt veremez olduğunu belirlemek için yapılan testler, performans testine örnek teşkil edebilir.

Örneğin; büyük bir veri tabanı yönetim sisteminin başarımı, arama ve sonucu gösterme işlemleri için gereken süredir. Gömülü bir sistemin performansı, insan katkısı olmadan yapmak zorunda olduğu işlemleri başarıyla yapmasıdır. Özellikle gerçek zamanlı sistemler için tanımlanmış olan zaman kısıtlarına uymak hem yazılım hem de donanım bileşenleri tarafından karşılanması gereken çok önemli isterlerdir. Bu isterlerin uygun şekilde karşılanıp karşılanmadığını görebilmek için tümleştirilen yazılım ve donanım üzerinde performans testleri yapılır. Performans testleri tüm test aşamalarında yapılabilir(birim,stress vs.

Yük Testi (Load Testing) :

Her bilgisayar sistemi belirli tür ve miktarda veriyi işlemek ve bunlara göre başka veriler üretmek için tasarlanır. Gerçek zamanlı sistemler ve kontrol sistemleri daha çok belirli kesmelere karşı bir işlem yaparak belirli bir tepkide bulunur. Veritabanı yönetim sistemleri ise çok miktarda veriyi saklama, bunlar üzerinde sorgu yapma ve rapor üretme gibi işlevler yürütür. İşte bu tür yoğun veri akışına sahip sistemler için yükleme testleri yapılır.

Yük testleri, sistemin sınırlarını zorlayarak en fazla ne kadar veri işleme kapasitesi olduğunu belirlemek, bu yükte davranışlarını kontrol etmek amacıyla yapılır. Hatta, bazı durumlarda, isterlerde belirtilen değerlerin de üzerine çıkılarak kaldırabilecek en fazla yükün ne olabileceği araştırılır.

Yükleme Testi (Loading Testing) :
Tipik bir sistemin yükleme testleri şunlardan oluşabilir:
* Veri hacmi testi: Sistemin yüksek miktarda veri ile yüklenmesi
* Veri debisi testi: Tüm girişlerin yüksek hızda veri ile yüklenmesi
* Kapasite testi: Sistemin bellek ve disk kullanımının zorlanması

Germe Testi (Stress Testing) :
Normal olmayan koşullarda, hem yazılım hem de donanımın ne şekilde davranacağını görmek üzere germe(stress) testleri yapılmalıdır.
Bu amaçla yapılacak testler şunlardır:
Sistemi normal olmayan miktarda öz kaynak gerektirecek şekilde zorlamak amacıyla yapılan testler.
* Yüksek hacim ve hızda veri girişi yapmak
* Beklenenden çok daha yüksek frekansta kesmeler yapmak
* Beklenen ve kullanılandan çok daha fazla bellek ve işlemci gücü gerektiren durumlar yaratmak.
Sistemin kaldırabileceği yük durumunda, ani etkilere verilecek tepki süresini ölçmek üzere yapılan testler

Kurtarma Testi (Recovery Testing) :
Bilgisayar sistemlerinin çoğunda bir hata durumunda kendini toparlayarak tekrar çalışmaya devam etmesi beklenir. Aşağıdaki yöntemler kullanılarak zararlar en aza indirilebilir. Yedekli yazılım mimarisinde, ana yazılım birimi çalışırken, yardımcı yazılım da aynı veya farklı bir donanım üzerinde paralel şekilde çalışır, fakat çıktı üretmez. Ana yazılımın çökmesi halinde bu yardımcı yazılım devreye girer. Hataya dayanıklı yazılım ise, herhangi bir nedenle bütünüyle çökmemek üzere, kendi kendini düzeltebilen modüller halinde geliştirilen yazılımlardır.
Kurtarma testi önce yazılımı sonra da donanımı çeşitli olası şekillerde bilinçli bir şekilde çökerterek sistemin kendini tekrar toplamasının denenmesi, isterlerin doğrulanması amacıyla yapılır. Bu kapsamda genelde şu testler yapılır:

* Yedekli yazılım mimarisinde, ana yazılımın devreden çıkartılması ve yardımcı yazılımın otomatik olarak devreye girmesi, bilgi işlemenin kayba uğramadığının kontrolü.
* Yeniden yazılım modülü başlatma yönteminde, çken modülün tekrar başlatılması ve çökmeden önceki durumunu kazanması
* Çökme sırasında kaybolma olasılığı olan verilerin tekrar alınması veya üretilmesi.
* İnsan katkısı gereken geri kazanma durumlarında, ortalama zamanın ölçülmesi, isterlere göre değerlendirilmesi.

Güvenlik Testi (Security Testing) :

Güvenlik testi, bir bilgi sisteminin verileri ve işlevselliğini korumak için tasarlanmış bir süreçtir. Herhangi bir bilgi sızıntısı olup olmadığını kontrol edilir. Sistemin tüm potansiyel açık kapıları ve zayıflıkları araştırılır.

Temel güvenlik testi çeşitleri şunlardır:

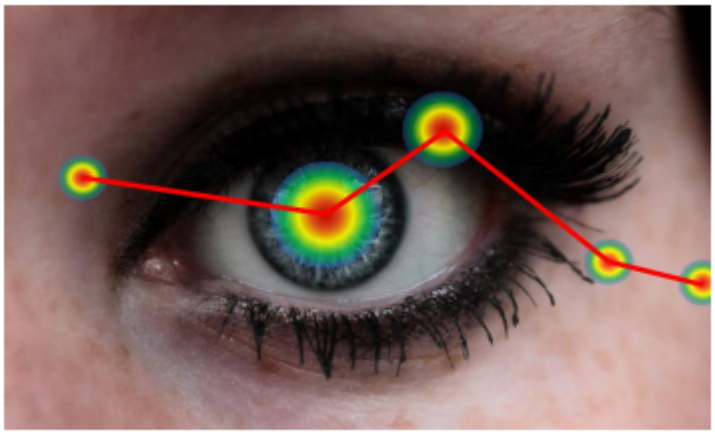
* Zafiyet taraması
* Penetrasyon Testi
* Risk Belirleme
* Güvenlik Denetimi
* Şifre Kırma

Taşınabilirlik Testi (Portability Testing) :
Taşınabilirlik testi, var olan bir yazılım bileşeni veya uygulamayı yeni bir ortamda test etme işlemidir. Uygulamanın diğer ortamlarda.
*Installability
*Combatibility
*Adaptability
*Replaceability
yetenekleri araştırılır. Sonuçlar rapor edilir. Uygulamalar şu ortamlar için test edilebilir:

* Donanım platformları(istemciler, sunucular, ağ bağlantı cihazları, giriş ve çıkış aygıtları)
* İşletim sistemleri (versiyonları ve servis paketleri dahil).
* Tarayıcılar (her bir sürümü dahil)

Kullanılabilirlik Testi (Usability Testing) :
Tasarımların veya ara yüzlerin kullanıcı ile buluşmasından önce tasarımın kullanılabilirliğini ölçmek amacıyla yapılan testlere denir. Sadece bir kullanıcının o an için hareketlerini gözlemlemek ile ilgilidir. Kullanılabilirlik testleri kullanılabilirlik problemleri hakkında bize bilgi verir ve kullanıcıların uygulama ile nasıl etkileşimde bulunduğuna bakar.
Kullanılabilirlik, 5 niteliksel özellik ile ölçülür.
* **Öğrenilebilirlilik**: Kullanıcılar, tasarımı ilk kullandıklarında yerine getirmeleri gereken görevleri kolaylıkla yapabiliyorlar mı?
* **Verimlilik**: Kullanıcılar, tasarımın çalışma şeklini öğrendikten sonra gerçekleştirecekleri işlemleri ne kadar hızlı yapabiliyorlar?
* **Memnuniyet**: Tasarımı kullanmak kullanıcıları duygusal anlamda mutlu ediyor mu, kullanıcılar tasarımı kullanırken kendini rahat hissediyorlar mı?
*__Hatırlanabilirlilik__: Kullanıcılar, bir süre tasarımı kullanmadıktan sonra tekrar kullanamaya başladıklarında tasarıma dair var olan bilgilerin ne kadarını hatırlayabiliyorlar?
* **Hatalar**: Kullanıcılar, ne kadar hata yapıyor ve bu hataları ne sıklıkta tekrarlıyorlar, hataları ne kadar hızlı yok edebiliyorlar?

Örnek olarak, bir web sitesinin kullanılabilirlik testini ele alalım. Göz izleme cihazı, kullanıcın nereye, ne kadar süre ve kaç kere baktığına, anlık ve geçmiş dikkatinin nerede yoğunlaştığına, niyetine ilişkin bilgi sağlar. Göz izleme cihazı göz bebeklerinin hareketlerini ve odaklanmalarını izleyerek odak noktaları haritası çıkarır. Bu bilgilere göre web sitemizi kullanıcının ilgisini çekebilecek ve daha kolay kullanımını sağlayacak şekilde düzenleyebiliriz.



Kabul Testi (Acceptance Testing) :
Çalıştırılmadan önce yazılımın son sınanmasıdır. Artık yapay veriler yerine gerçek veriler kullanılır. Bu sınama türü;
*Alfa sınaması
*Beta sınaması olarak çeşitlenir.
Alfa Testi (Alpha Testing) :
Geliştiricinin kendi yerinde müşteri tarafından yapılır. Geliştirici bu testleri gözlemleyerek gerçek kullanım hakkında bilgi sahibi olmaya çalışır; kusur buldukça not alır ve düzenleme işlemlerini yürütür. Alfa testlerinin en önemli özelliği, denetim altındaki bir ortamda, asıl kullanıcılardan biri tarafından yapılıyor olmasıdır.

Beta Testi (Beta Testing) :
Birçok kullanıcının kendi ortamında yapılır. Geliştirici genellikle bu testlere katılmaz; yalnızca belirli aralıklarla sonuçları ve yorumları alır. Bu testin özelliği de geliştirici tarafından kontrol edilemeyen gerçek uygulama ortamı koşullarında yazılımın denenmesidir. Beta testi sonunda geliştirici, bulunan kusurları düzelterek tüm kullanıcılar için yeni bir sürüm çıkartır.

İlgili

SAP-MM Modülü -MM17 Toplu Malz Bakımı (https://oktayolgun.com.tr/2021/...mm-modulu-mm17-toplu-malz-bakimi/)
Mart 5, 2021
"SAP" içinde

Siber Ölüm Zinciri Nedir. (https://oktayolgun.com.tr/2021/...olum-zinciri-nedir/)
Haziran 15, 2021
"GÜNCEL HABER" içinde

NTLM ve KERBEROS (https://oktayolgun.com.tr/2021/...ve-kerberos/)
Temmuz 5, 2021
"GÜNCEL HABER" içinde

Posted In
SECURİTY (https://oktayolgun.com.tr/category/security/), Uncategorized (https://oktayolgun.com.tr/category/uncategorized/)

Tags
Security (https://oktayolgun.com.tr/tag/security/)
Security (https://oktayolgun.com.tr/tag/security/)
About The Author
oktayolg (https://oktayolgun.com.tr/author/oktayolg/)
Network , Cisco ve SAP Uzmanı Elektronik Haberleşme Mühendisi İTU Mezunu
Comments
Leave a response (https://oktayolgun.com.tr/2020/05/11/171/)
‹ (https://oktayolgun.com.tr/2020/05/11/agdaki-ana-bilgisayarlari-hizla-bulmak/)
›

Top Five
Heat Index
SORT