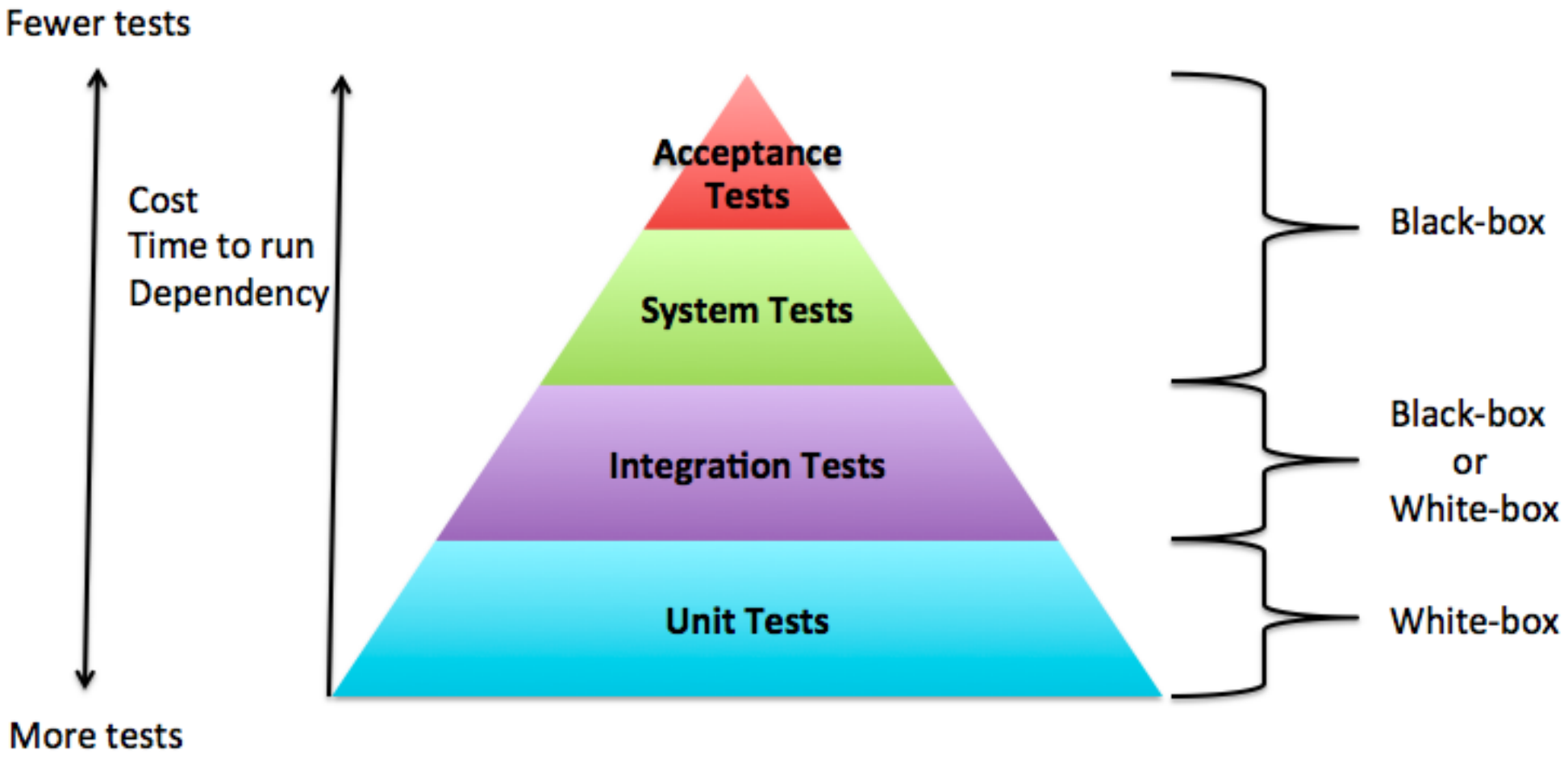
SOFTWARE TESTING LIFE CYCLE

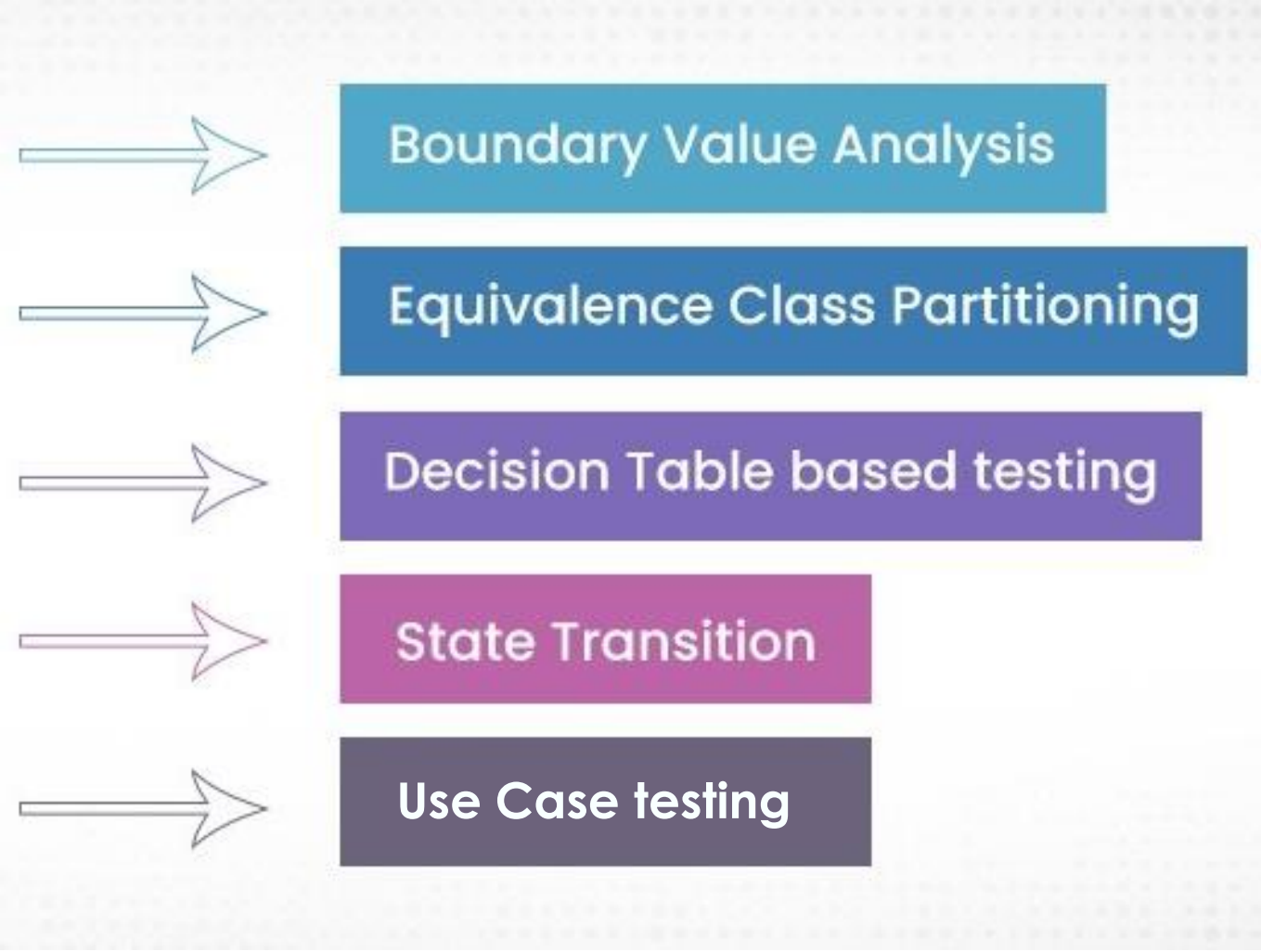
(Yazılım Testi Yaşam Döngüsü)

20.08.2022

Test Piramidi – Test Hiyerarşisi



BLACK BOX (KARA KUTU) TEST TENİKLERİ



Sınır Değerleri Analizi

Denklik Paylarına Ayırma Tekniği

Karar Tablosu

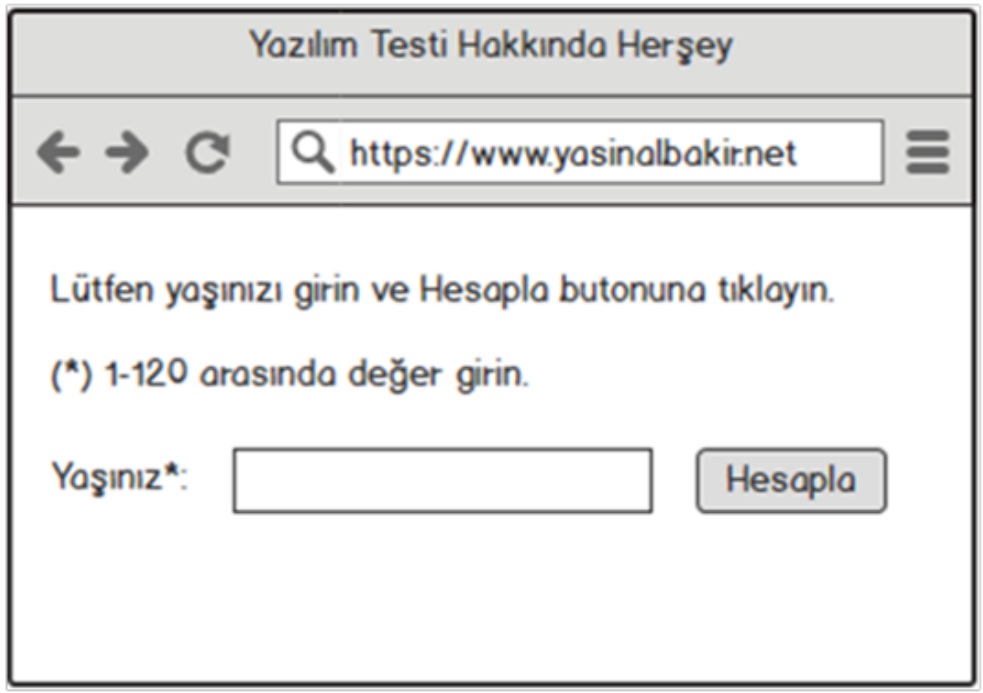
Durum Geçiş Tablosu

Kullanım Durumları Testi

BLACK BOX (KARA KUTU) TEST TENİKLERİ

1- Denklik Paylarına Ayırma - Equivalence Partitioning (EP)

Denklik paylarına ayırma test tekniğinin amacı benzer özelliklere sahip test senaryolarını gruplandırılarak daha az test senaryosu yazılmasıdır. Bu tekniğe göre aynı çıktıyı veren test girdi grubu için bir tane test senaryosu yazılması yeterli görülür. Bu test tekniği tüm test seviyelerinde uygulanabilir



Geçerli ve geçersiz test girdilerinin (denklik sınıflarının) aşağıdaki gibi olduğunu söyleyebiliriz.

- Geçersiz Aralık: 0 ve 0'dan küçük sayılar
- Geçerli Aralık: 1-120 arasındaki sayılar
- Geçersiz Aralık: 121 ve 121'den büyük sayılar

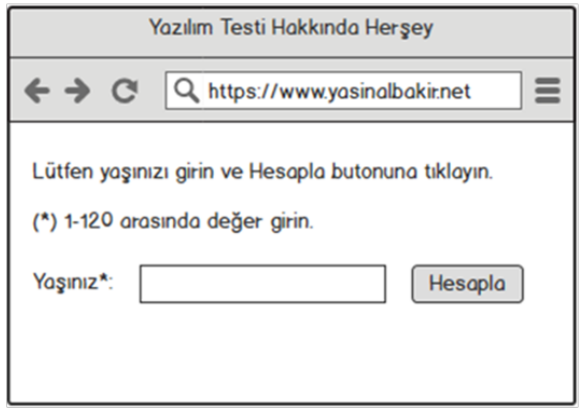
Denklik sınıflarının doğru belirlenmesi test tasarımınızı ve dolayısıyla yapacağınız testlerin kalitesini olumlu şekilde etkileyecektir.

BLACK BOX (KARA KUTU) TEST TENİKLERİ

2 - Sınır Değer Analizi - Boundary Value Analysis

Sınır değer analizi test tekniği tüm test seviyelerinde uygulanabilir. Uygulaması kolay hata bulma becerisi yüksektir. Bu teknik özellikle denklik paylarına ayırma test tekniği ile kullanılırsa daha etkili sonuçlar elde edilebilir. Yine doğum tarihi hesaplama uygulaması üzerinden örnek vermek gerekirse;

	Formül	**Test Girdisi**
Geçersiz Aralık:	Alt sınır değeri – 1	0
Geçerli Aralık:	Alt sınır, Alt sınır + 1 ve Üst Sınır -1, Üst Sınır	1,2 – 119,120
Geçersiz Aralık:	Üst Sınır + 1	121

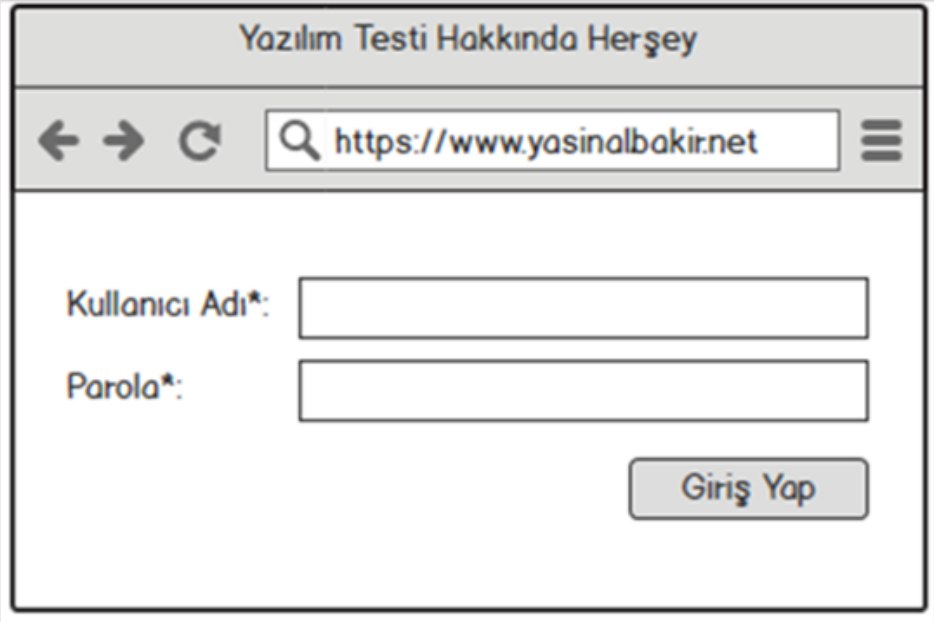


Sınır değer analizi tekniğinde görüldüğü üzere bir sınır değerinin testi için 3 farklı test senaryosu yazılabiliyor.

BLACK BOX (KARA KUTU) TEST TENİKLERİ

3- Karar Tablosu Testi - Decision Table Testing

Karmaşık iş kurallarına sahip sistemlerin test edilmesinde kullanılan test tekniğidir. Karar tablosu test tekniğinin en büyük avantajı test sırasında gözden kaçabilecek olasılıkların net bir şekilde listelenerek gözden kaçırma riskinin en düşük seviyelere indirilmesidir. Gelin bu test tekniğini aşağıdaki örnekle daha iyi anlamaya çalışalım.



Durum	**Kural 1**	**Kural 2**	**Kural 3**	**Kural 4**
Kullanıcı adı doğru mu ?	Evet	Evet	Hayır	Hayır
Parola doğru mu ?	Evet	Hayır	Evet	Hayır
Aksiyon				
Oturum açma başarılı	Evet			
Oturum açma başarısız		Evet	Evet	Evet

BLACK BOX (KARA KUTU) TEST TENİKLERİ

4- Kullanım Durum Senaryosu Testi - Use Case Testing

Kullanım durum senaryoları sistem ile aktör arasındaki etkileşimi göstermek için yazılan senaryolar topluluğudur. Kullanım durum senaryosu testinin amacı sistemi baştan sona her bir senaryonun teker teker bütün sistemi kapsayacak şekilde test edilmesidir. Kullanım durum senaryolarında ana senaryo ve alternatif senaryolar bulunur. Ana senaryolarda sistemin yerine getirmesi gereken adımlar yer alır. Alternatif senaryolarda ise sistemin uç noktaları için test adımları yazılır.

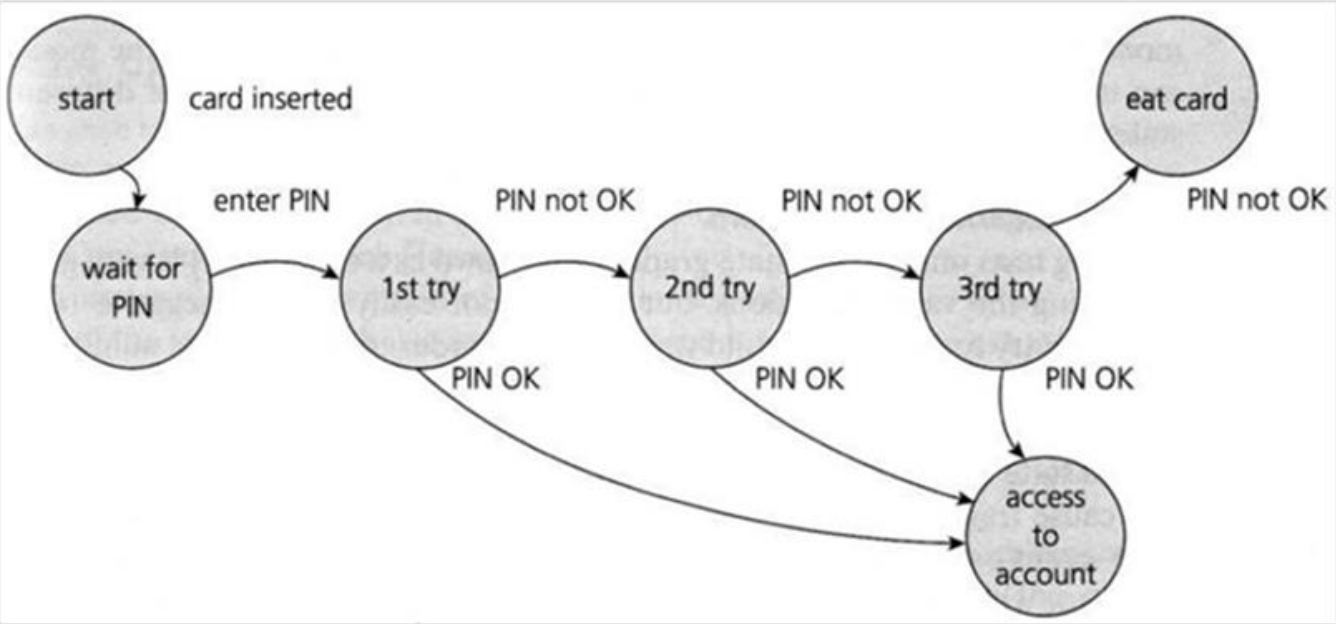
Örneğin bir kullanıcı tanımlama ekranında ana senaryo ekranda yer alan tüm zorunlu alanların doldurularak kaydedilebilmesi iken alternatif senaryoda ekranda yer alan bir veya birden fazla zorunlu alanın boş bırakılarak sistemin verdiği tepkinin ölçülmesi olabilir.

Ekran Adı	Kullanıcı Tanımlama ve Güncelleme
Tanım	Yetkili kullanıcının kullanıcı tanımlama ve güncellemesi amaçlanmıştır.
Aktörler	Yetkili Kullanıcı
Gereksinim Numarası	SRS-KM-1
Ön Koşullar	Yetkili kullanıcı sisteme giriş yapmış olmalıdır.
Ana Senaryo	1. Yetkili kullanıcı ekranda yer alan aşağıdaki alanları doldurur/seçer; a. Adı* b. Soyadı* c. TCKN* d. E-Posta* e. Cep Telefonu f. Adres 2. Yetkili kullanıcı, **Kaydet** butonuna basar. 3. Sistem, kullanıcı kaydetme işleminin başarıyla tamamlandığına dair bilgilendirme mesajı verir.
Alternatif Senaryo	1a) Yetkili kullanıcı ekranda yer alan zorunlu alanları boş bırakarak kaydet butonuna basar. 2a) Yetkili kullanıcı sistemde tanımlı olan kullanıcının TCKN bilgisini kullanarak yeni kullanıcı tanımlamaya çalışır. 2b) Yetkili kullanıcı sistemde tanımlı olan kullanıcının E-Posta bilgisini kullanarak yeni kullanıcı tanımlamaya çalışır.
Son Koşullar	3a) Sistem kaydetme işleminden sonra işlem kaydını (log) veri tabanına kaydeder.

BLACK BOX (KARA KUTU) TEST TENİKLERİ

5- Durum Geçiş Testi - State Transition Testing

Durum geçiş testinin amacı belli iş kurallarına bağlı olarak şartların oluşmasına ve bir durumdan diğerine geçilerek bir noktada sonlanması durumunu test etmektir. Bu tür sistemlerde bu durumlar durum geçiş diyagramı ile gösterilir. Durum geçiş test tekniğini aşağıdaki örnek ele alalım.



Yanda yer alan ATM örneğinde ilk olarak PIN kodu bekleniyor daha sonra girilen PIN kodu kontrol ediliyor eğer PIN kodu doğruysa kullanıcı hesabına erişebiliyor 3 hatalı PIN kodu girişinde ise ATM kartı yutuyor.

Durum geçiş test tekniği genellikle gömülü sistemlerin testlerinde kullanılmaktadır.

Kara kutu test teknikleri ile projelerinizde daha efektif test senaryoları tasarlayabilirsiniz.

BUG LIFE CYCLE – TEMEL KAVRAMLAR

ERROR :

Koddaki Sorun hatalara yol açar; bu, geliştiricinin gereksinimi yanlış anlaması veya gereksinimin doğru tanımlanmaması nedeniyle geliştiricinin kodlama hatası nedeniyle bir hata meydana gelebileceği anlamına gelir. Geliştiriciler **error terimini** kullanır . Error sonucu olarak ortaya bug ve defect kavramı çıkar.

BUG / DEFECT?

Application da Gereksinimlere uygun olmayan her hangi bir davranis (**Unexpected behaviour of an application)**

Bug ve Defect kavramları her ne kadar aynı görünse bile birbirinden farklı kavramlardır. Bu kavramlarla alakalı farklı dokümanlarda farklı şekillerde karşılaşabilirsiniz. Ama piyasada kabul gören anlamı ile şu şekilde belirtebiliriz

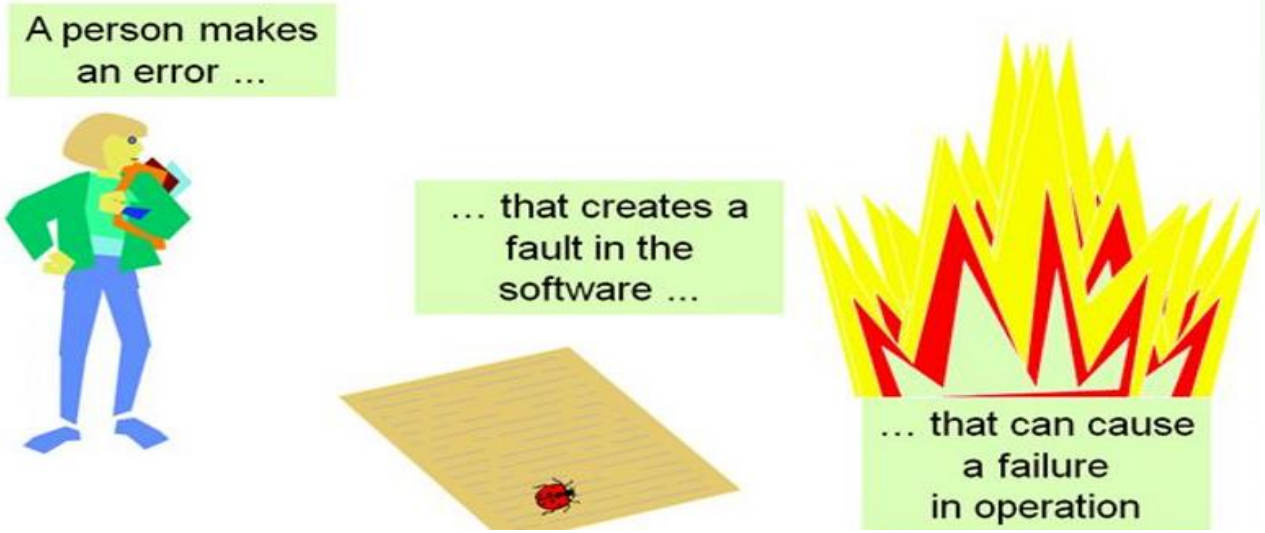
BUG LIFE CYCLE – TEMEL KAVRAMLAR

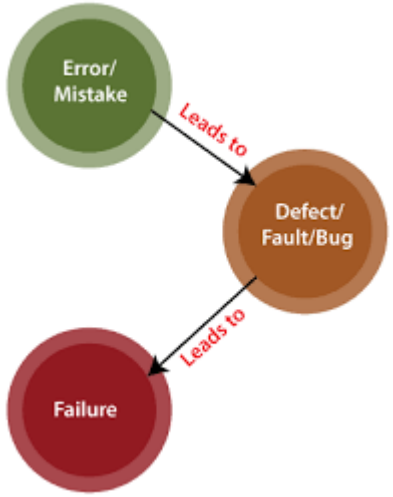
BUG / DEFECT?

Defect : Uygulamanın geliştirme aşamasında karşımıza çıkan kusurlara denir. Bunu developerda bulabilir , testerda. Önemli olan hangi aşamada bulunmuş olmasıdır

Bug: Uygulama tamamlandıktan sonra bulunan hatalardır. Bu hataları testerlarda bulabilir yada son kullanıcı da bulabilir. Bu tür hatalara bug denir

Failure : bir hatanın ortaya çıkardığı sonuç olarak tanımlanır





BUG LIFE CYCLE

- Yazılım testinde Kusur Yaşam Döngüsü veya Hata Yaşam Döngüsü, kusur veya hatanın ömrü boyunca geçtiği belirli durumlar kümesidir.
- Kusur yaşam döngüsünün amacı, tespit edilen kusurun mevcut durumunu raporlamak, ilgili kisilere atamak ve **Bug Defect (Hata Duzeltme)** sürecini sistematik ve verimli hale getirmektir

!!! TESTER OLARAK BUG BULDUĞUMUZDA NE YAPMALIYIZ !!!

BUG LIFE CYCLE

!!! TESTER OLARAK BUG BULDUĞUMUZDA NE YAPMALIYIZ !!!

- Bir bug bulduğumuzda öncelikle bunun bug olduğundan emin olmamız gerekiyor.
- Bunun için testimizi tekrarlamalı, gerekirse başka tester var ise onun fikrini almamızda fayda var. Test Leade bilgi verilebilir.
- Eğer developerlar ile aynı ortamda çalışılıyorsa, developer durumdan haberdar edilebilir. Developer bunun bug olmadığını, bug olduğunu ya da önemsiz bir durum olduğunu ifade edebilir.
- Developerin dönüşüne göre hareket edilir.

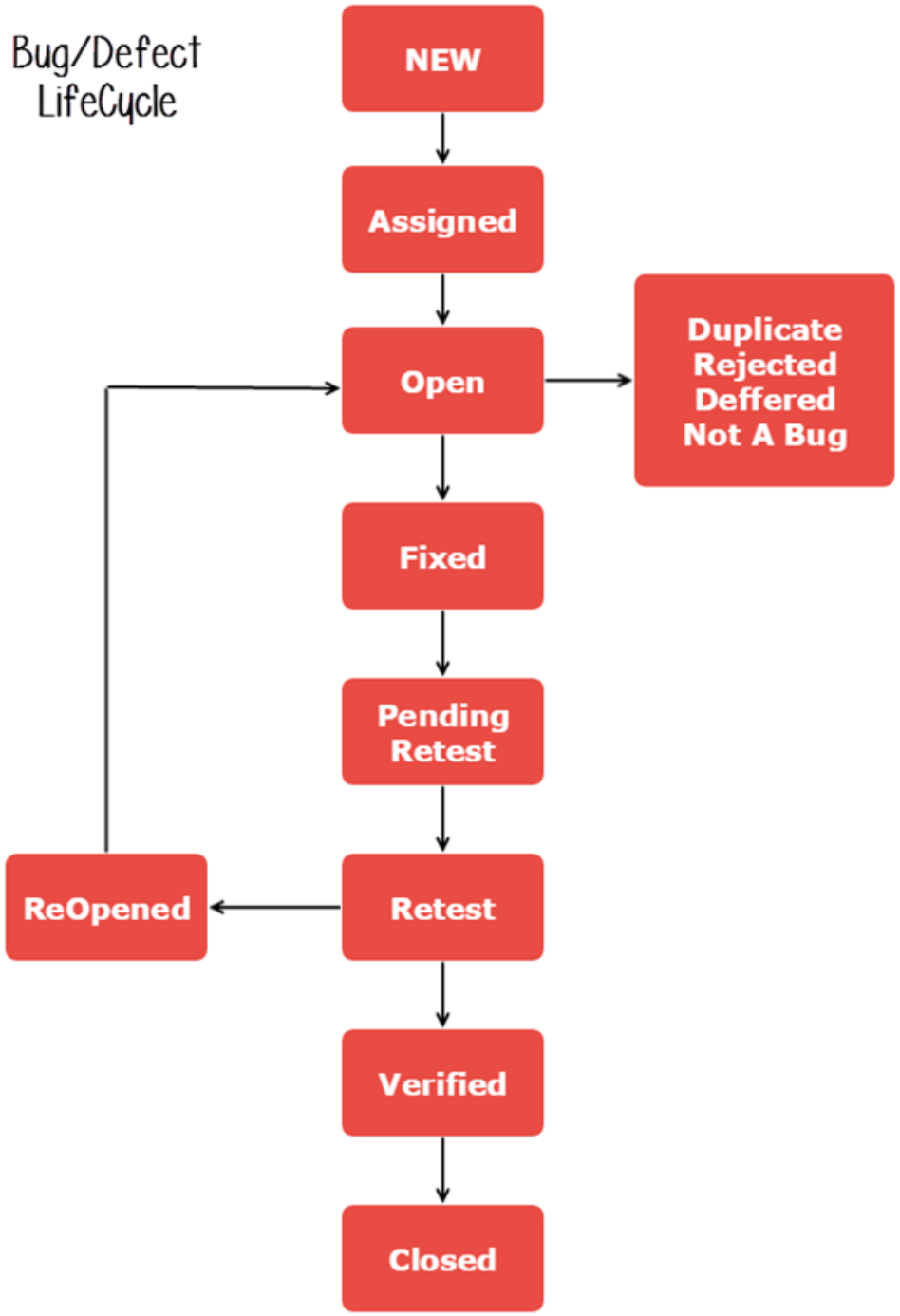
BUG LIFE CYCLE

New (Yeni) : Bir hata ilk kez yayınlandığında, durum Yeni olur. (Jira, Alm, VersionOne gibi boardlarda..)

Assigned (Atandi) : Hata post edilince, test lead veya Proje lead, ilgili Developer'a "atanmış" olarak atayacaktır. (Meşru bir hata olup olmadığı ile alakali Developerin kesinlikle bir girdisi olacaktır)

Open (Acildi) : Developer bu aşamada düzeltmek için hatayi analiz eder ve uzerinde çalışır.

Fixed (Onarildi) : Developer gerekli kod güncellemelerini yaptığında ve hatanın düzeltilip düzeltilmediğini doğruladığında, hata durumunu "Fixed" olarak değiştirebilir ve hata test ekibine aktarılır



BUG LIFE CYCLE

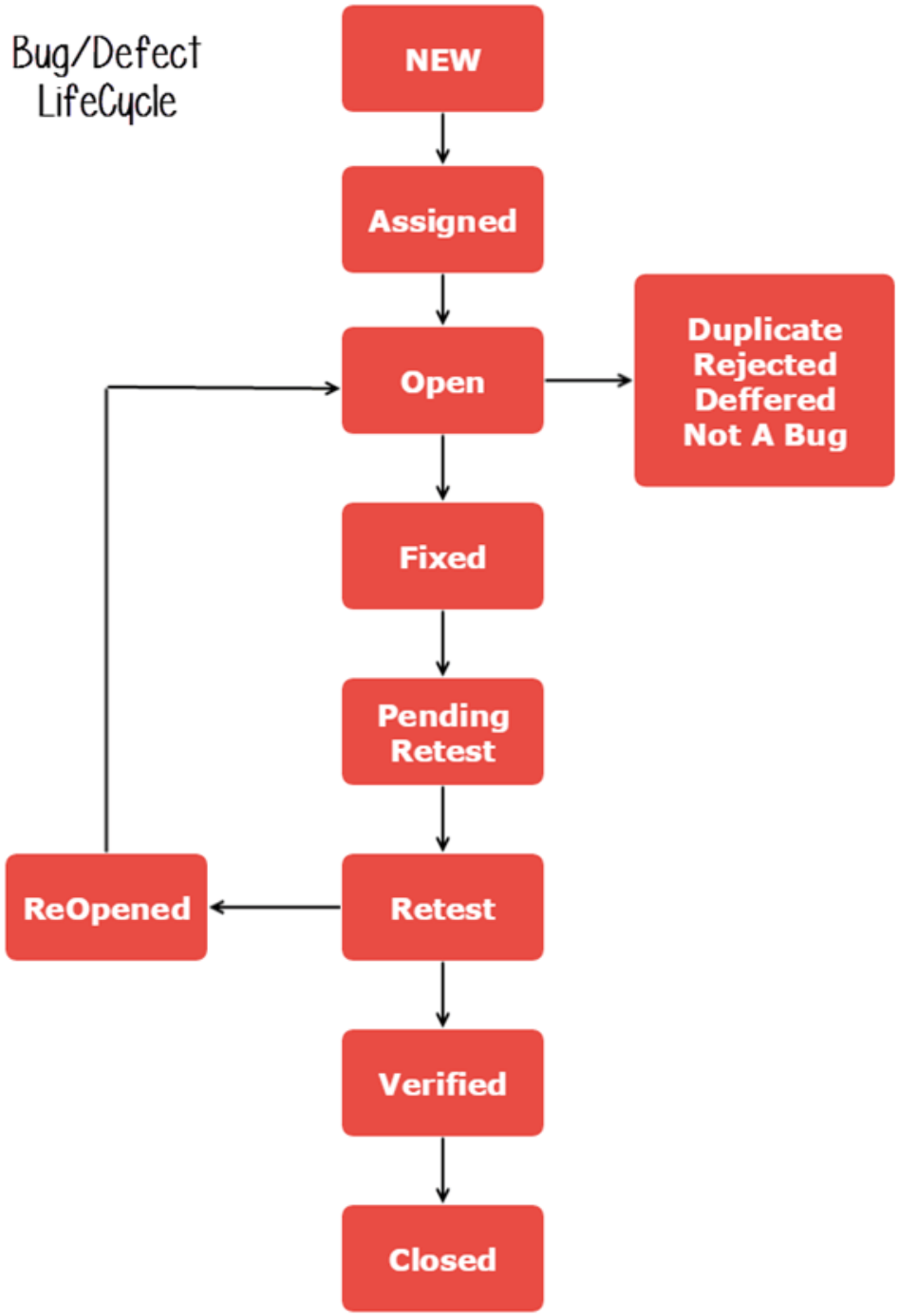
Pending retest (Tekrar Test Icin Bekliyor) : Arızayı giderdikten sonra Developer, yeniden test için bu özel functionalityi deploy eder. Burada testler test aşamasında beklemede olur. Ve durum tekrar test etmeyi gerektirir.

Retest (Tekrar Test) : Bu aşamada, tester işlevselliği farklı test verileriyle tekrar test eder ve buldukları defect ilgili her şeyi kapsadığından emin olur.

Verified (Onaylandi) : Tester doğrulama yaptıktan ve hata yazılımda görünmediğinde, hatanın düzeltildiğini onaylar ve durumu “Doğrulandı” olarak değiştirir.

Reopen(Tekrar Acildi) : Hata Developer tarafından düzeltildikten sonra bile hata hala mevcutsa, tester durumu "Yeniden Açıldı" olarak yapar. Bu yüzden hata aynı süreçten bir kez daha geçer.

Closed(Kapandı) : Hata giderildikten sonra tester tarafından test edilir. Tester yazılımda defect bulunmadığından emin olursa, hatanın durumunu “Kapalı” olarak değiştirir. Bu, hatanın düzeltildiği, test edildiği ve onaylandığı anlamına gelir.



BUG LIFE CYCLE

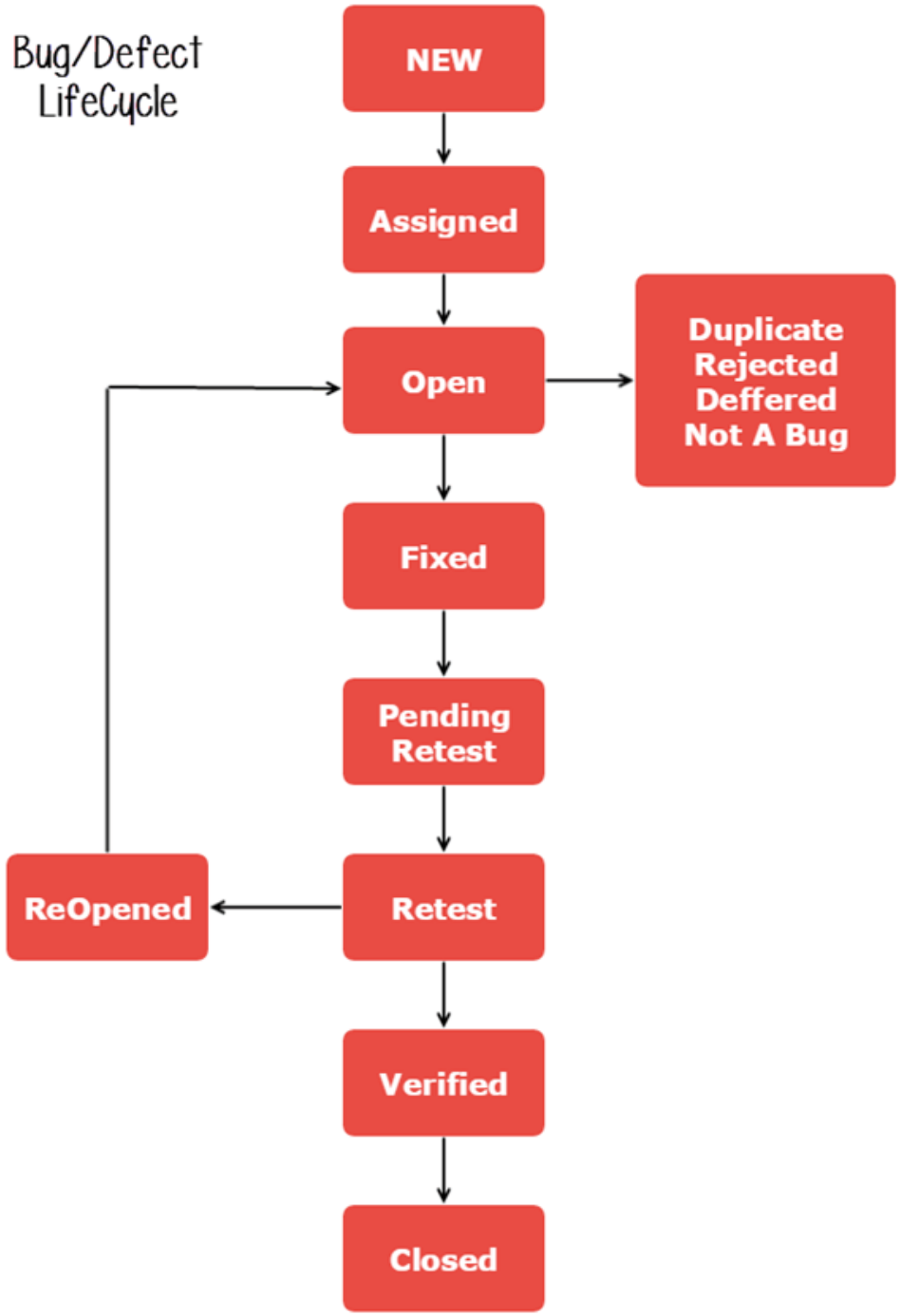
Bug Ticket olusturuldugunda Developer'larin cevaplari

Rejected (Reddedildi) : Kusur Developer tarafından meşru bir kusur olarak kabul edilmezse Developer tarafından 'Reddedildi' olarak işaretlenir.

Duplicate (Tekrar) : Developer hatayi diğer diger defectlerle aynı bulursa, hata durumu Developer tarafından "cift" olarak değiştirilir.

Deferred (Ertelendi) : Developer, hatanın çok önemli bir öncelik olmadığını ve sonraki sürümlerde giderilebileceğini düşünüyorsa, hatanin durumunu 'Ertelenmiş' olarak değiştirebilir.

Not a Bug (Bug Degil) : Kusurun uygulamanın işlevselliği üzerinde bir etkisi yoksa, kusurun durumu 'Hata Değil' olarak değiştirilir.



BUG LIFE CYCLE

Bug Raporu (Bug Report)

Hata raporlama işlemi proje yönetim aracı üzerinde gerçekleştirilebilirken, aşağıdaki başlıklardan oluşan bir dokumanda oluşturulabilir. Bu dokuman sayesinde geliştirici hatayı daha rahat anlayabilir.

Başlık (Title) : Hata başlığı diğer hatalarda ayırt etmek için yazılır. Hata başlığı kısa ve öz olmalıdır. Örnek vermek gerekirse *"Kullanıcı tanımlama ekranında kaydet butonuna basıldığında hata veriyor."*

Atama (Assigned To) : Bulunan hatanın hangi geliştiriciye veya takım liderine atanacağı bu alandan seçilmelidir.

Öncelik (Priority) : Yazılım Test Planın'da hatalar önceliklerine göre derecelendirilmelidir. Bulunan hataların öncelik değerleri plana göre doğru şekilde girilmelidir.

Önem Derecesi (Severity) : Bulunan hatalar önem derecesine göre gruplandırılmalıdır. İlgili geliştirici hataları önem derecesine göre çözmelidir.

Hata Kaynağı (Root Cause) : Bulunan hatanın kaynağı belirtilmelidir. Bu bilgi test faaliyetleri sonucunda ölçüm ve analiz için kullanılabilir.

BUG LIFE CYCLE

Bug Raporu (Bug Report)

Adımlar (Repro Steps) : Bulunan hatanın ilgili geliştirici tarafından tekrar simüle edilebilmesi için hatanın adım adım yazılması gerekmektedir. Bunun için aşağıdaki örneği verebiliriz:

a. Kullanıcı tanımlama sayfasına gidilir.

b. Ekranda yer alan tüm zorunlu alanlar doldurulur.

c. Kaydet komutu verilir.

- **Beklenen Sonuç: Kullanıcı tanımlama işleminin başarıyla tamamlandığına dair bilgilendirme mesajı verilmesi gerekir.**
- **Gerçekleşen Sonuç: NullPointerException hata mesajı alındı.**

Ekler (Attachments) : Bulunan hatanın ekran görüntüsünü veya video kaydını alarak bu alana yüklenmesi test ekibinin hatayı geliştirme ekibine daha kolay aktarabilmesini sağlayacaktır.

	A	B	C
1	**Category**	**Label**	**Value**
2	Bug ID	ID number	#123
3		Name	CART - Unable to add new item to my cart
4		Reporter	Mike A
5		Submit Date	03/04/16
6	Bug overview	Summary	When my cart contains one item, I am unable to add a second item via the add to cart button on a product page
7		URL	www.example.com/product/abc
8		Screenshot	www.example.com/screenshot123
9	Environment	Platform	Macintosh
10		Operating System	OS X 10.12.0
11		Browser	Chrome 53
12	Bug details	Steps to reproduce	add one item to cart > go to product abc via the search bar > add new item to cart via "add to cart" button (see screenshot) > go to cart
13		Expected result	The cart should contain 2 items
14		Actual result	The cart contains only 1 item
15		Description	/
16	Bug tracking	Severity	Major
17		Assigned to	/
18		Priority	High
19	Notes	Notes	/

BUG LIFE CYCLE

Bug Tracker

All changes saved

Bugs | File Edit View Insert Format | Share

Arial 10

	A - BUG ID (Autogenera	B - Type	C - Found In	D - Feature	E - Description	F - Priority	G - Screenshots & Atta
1	**Bug Tracking**						
2	Track bugs and enhancements, with attached screenshots and mockups, status, owners, release assignments, and more.						
3	Use CTRL+V to copy screenshots from your clipboard into Attachment cells.						
4							
5	BUG ID (Autogenerated)	Type	Found In	Feature	Description	Priority	Screenshots & Attach
6	BUG-000001	Bug	Production	User Profile	When user uploads a photo, placeholder image doesn't get replaced until the user refreshes. This should happen right away as soon as the upload is completed.	High	
7	BUG-000002	Bug	Stage 1	Marketplace	Marketplace listings are not appearing in reverse chronological order by default as the should be	Critical	
8	BUG-000003	Bug	Local	Registration	Pinch gestures not working in search results pages on Android	Medium	
9	BUG-000004	Investigation	Production	Billing	Credit card approval notifications delayed by longer than expected	Critical - High Priority	
10	BUG-000005	Regression	Stage 2	Page Editor	Line breaks lost in embedded code	High	
11	BUG-000006	Bug	Production	User Profile	Settings option shows as highlighted even when mouse is not hovered over it	Medium	
12							