



YMH412-Yazılım Kalite Güvencesi ve Testi

Yazılım Testleri

Prof. Dr. Resul DAŞ
Fırat Üniversitesi Yazılım Mühendisliği Bölümü

Bölüm-3

İçindekiler

1	Yazılım Testi.....	3
2	Türkiye Yazılım Kalite Raporu.....	8
3	Testin Amaçları.....	13
4	Doğrulama Ve Geçerleme	15
5	Yazılım Hataları.....	23
6	Testi Kim Yapar?.....	32
7	Yazılım Geliştirme Metodolojilerinde Test	35

1. Yazılım Testi

➤ **Yazılım testi**, yazılımın daha önce tanımlanmış teknik ve işlevsel gereksinimleri karşılayıp karşılamadığının ve yazılımın beklendiği gibi çalışıp çalışmadığının kontrolüdür. **Yazılım test süreci** de temel olarak elde edilen ürünün beklenen kalitede olduğunu belirlemek, değilse istenilen kaliteye ulaştırılmasını sağlamayı amaçlayan bir süreçtir.



1. Yazılım Testi

- Bilgisayar sistemleri hatalar barındırır. Bu hatalar insan (geliştirici, analist vb.) ve donanım kaynaklı olabilir ve hatalar maddi ve/veya prestij kayıplarına neden olabilir. Yazılım test sürecinin proje döngüsüne katılmasıyla bu kayıplar en aza indirebilir. **Burada unutulmaması gereken şey mükemmel yazılım olmadığı ve bir yazılımın asla %100 test edilemeyeceğidir.**

1. Yazılım Testi

- Bir yazılımı %100 oranında test etmeye çalışmak hem maddi gerekçelerle hem de pratik olarak mümkün olmadığından, yazılımın beklendiği gibi çalıştığını gösterebilecek sınırlı sayıda ancak özenle seçilmiş bir test kümesinin belirlenmesi ve bu test kümesi üzerinden test yapılması daha mantıklıdır.
- Yazılım testi başka bir ifadeyle bir yazılımın **doğrulanması ve geçerlenmesi** süreci olarak da tanımlanabilir.

1.1. Yazılım Testinin Gelişimi



- Yazılım testi ilk ortaya çıktığı sıralarda yalnızca hata ayıklama amacıyla yapılmaktaydı. Sonra testler yazılımın doğru çalıştığını göstermek amacıyla yapılmaya başlandı. Daha sonraları testlerin yapıcı olmaktan çok, yıkıcı bir şekilde yapılmasının daha iyi sonuçlar verdiği görüldü.
- 1980'li yıllardan sonra daha kurallı geliştirme teknikleri kullanılmaya başlandığından tüm geliştirme sürecini içeren aşamalı testler kullanıldı. Günümüzde de bu yöntem yanında hataları önlemeye yönelik testler yapılmalıdır.

1.1. Yazılım Testinin Gelişimi

History of Software Testing



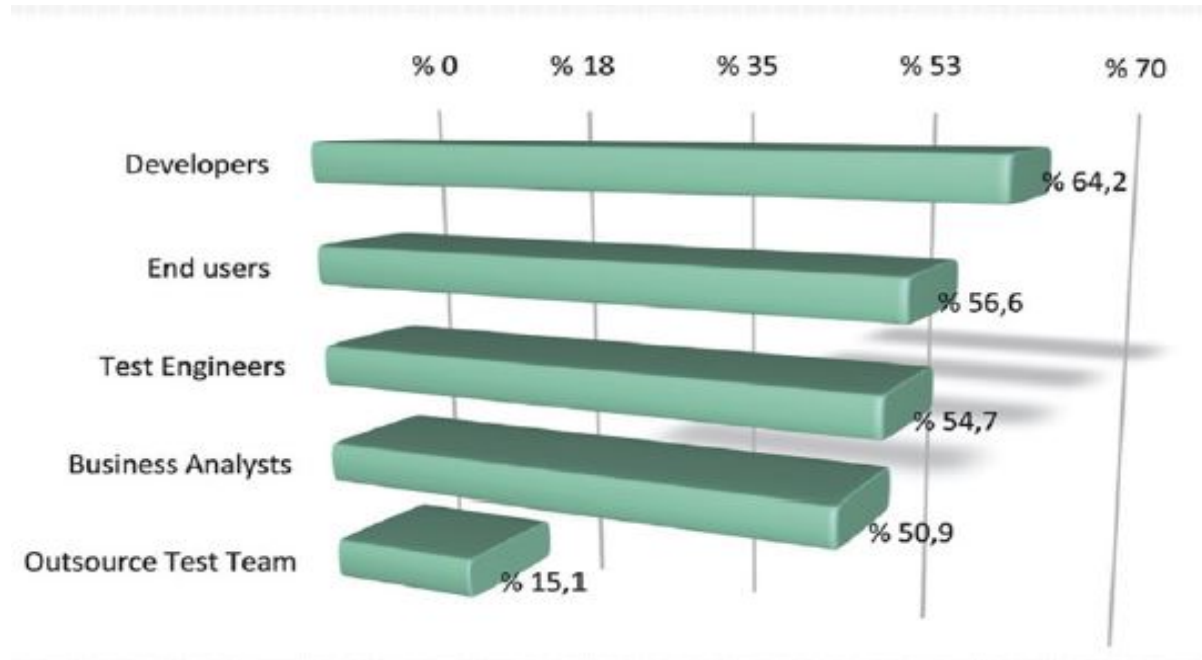
2. Türkiye Yazılım Kalite Raporu

- Veriler Türkiye Yazılım Kalite Raporu 2011-2012' den alınmıştır. 20 sorudan ve 4 başlıktan oluşan anketi 250 yazılım geliştirme firmalarının ilgili IT direktör, Kalite Müdür, İş analisti, Proje Yöneticisi gibi kişilerin cevaplanması istenmiştir. Cevaplar rapor halinde sunulmuştur.



2. Türkiye Yazılım Kalite Raporu

➤ Firmanızda test işleminden kimler sorumludur?

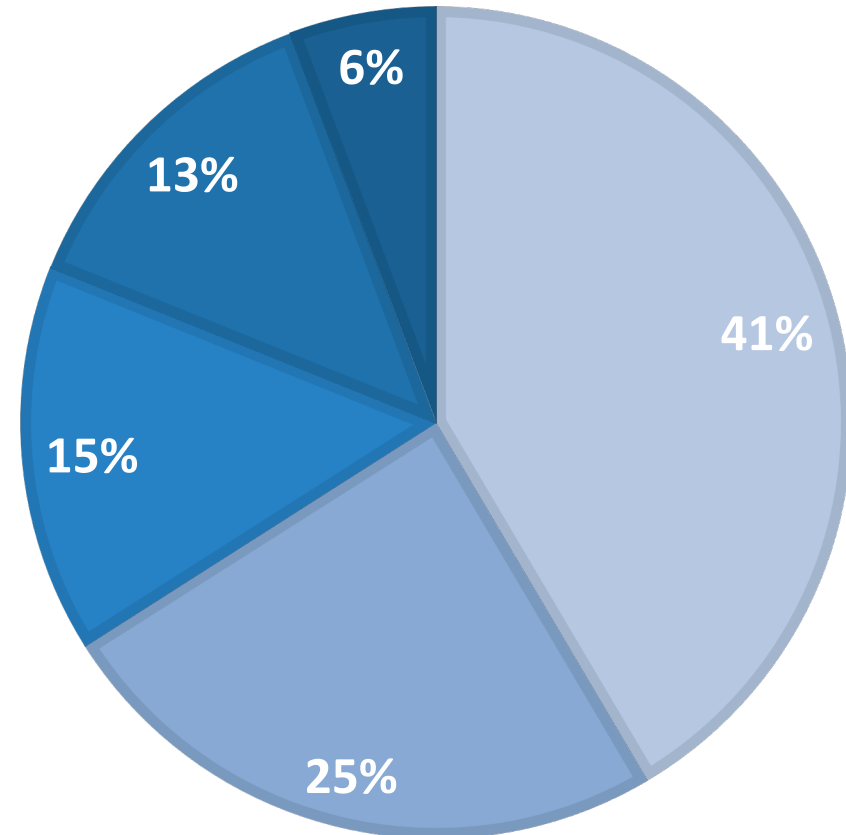


Soruya birden fazla cevap verebilme imkanı vardır.

2. Türkiye Yazılım Kalite Raporu

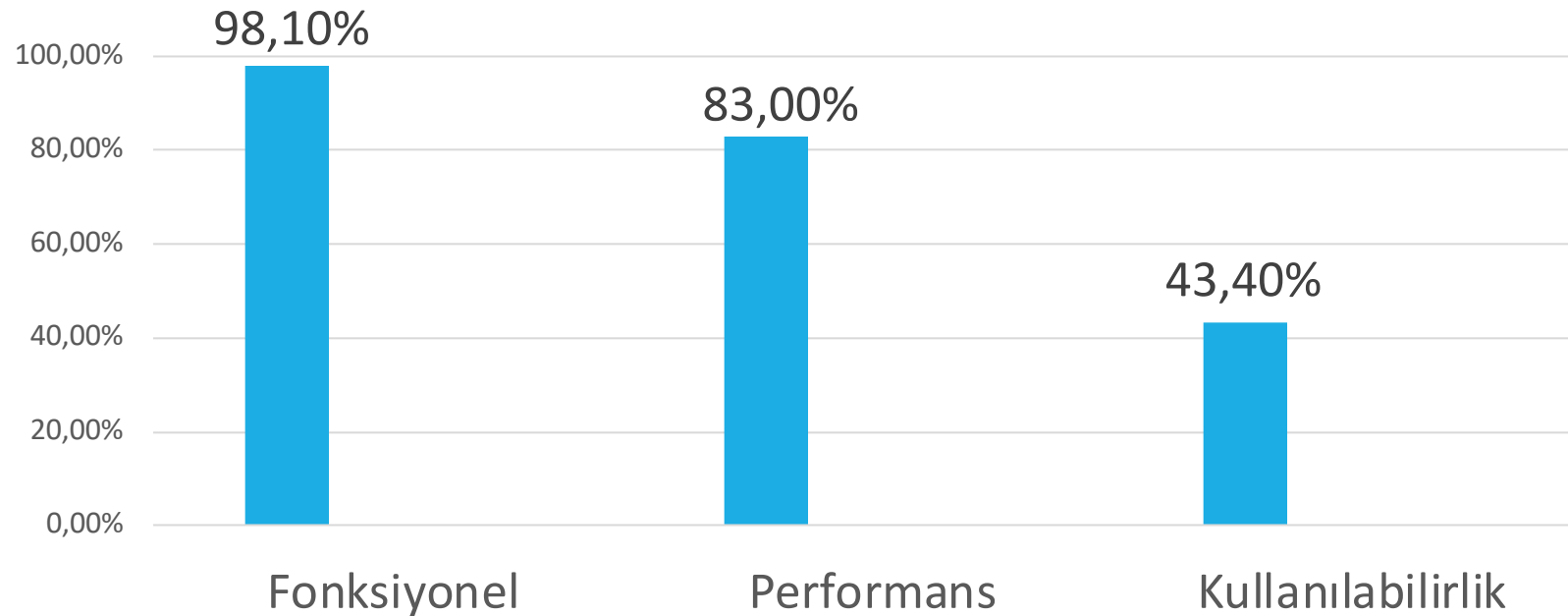
➤ İşletmenizde kaç adet test elemanı var?

■ Hiç Yok ■ <5 ■ <10 ■ <50 ■ 50 ve fazlası



2. Türkiye Yazılım Kalite Raporu

➤ En çok gerçekleştirdiğiniz test türleri nelerdir?

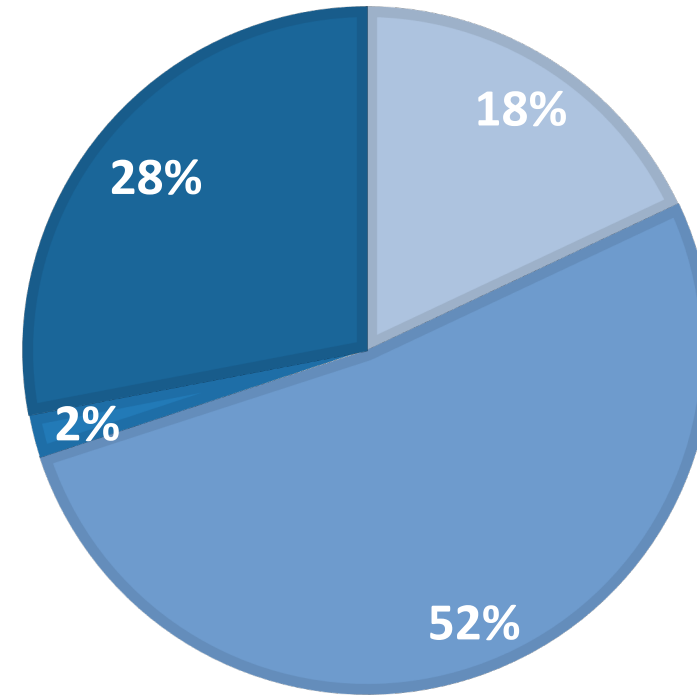


Soruya birden fazla cevap verebilme imkanı vardır.

2. Türkiye Yazılım Kalite Raporu

➤ Çalışanlarınızın test sertifikasına sahip olmaları ne kadar önemlidir?

■ Çok önemli ■ Önemli
■ Önemli değil ■ Fikrim yok



3. Testin Amaçları

- Müşteriye sunmadan önce **ürün kalitesinden** emin olmak,
- Yeniden çalışma ve geliştirme için **masrafları azaltmak**,
- Geliştirme işleminin erken aşamalarında hataları saptayarak ileri aşamalara yayılmasını önlemek, böylece **zaman ve maliyetten tasarruf** sağlamak,
- **Müşteri memnuniyetini** arttırmak ve izleyen siparişler için zemin hazırlamak.

3.1. Yazılım testinde neler test edilmektedir?

- İş gereksinimleri
- İşlevsel tasarım gereksinimleri
- Teknik tasarım gereksinimleri
- Düzenleyici gereksinimler
- Yazılımın kaynak kodu
- Ortakların standartları
- Donanım yapılandırılması ve dil farklılıkları

4. Doğrulama Ve Geçerleme

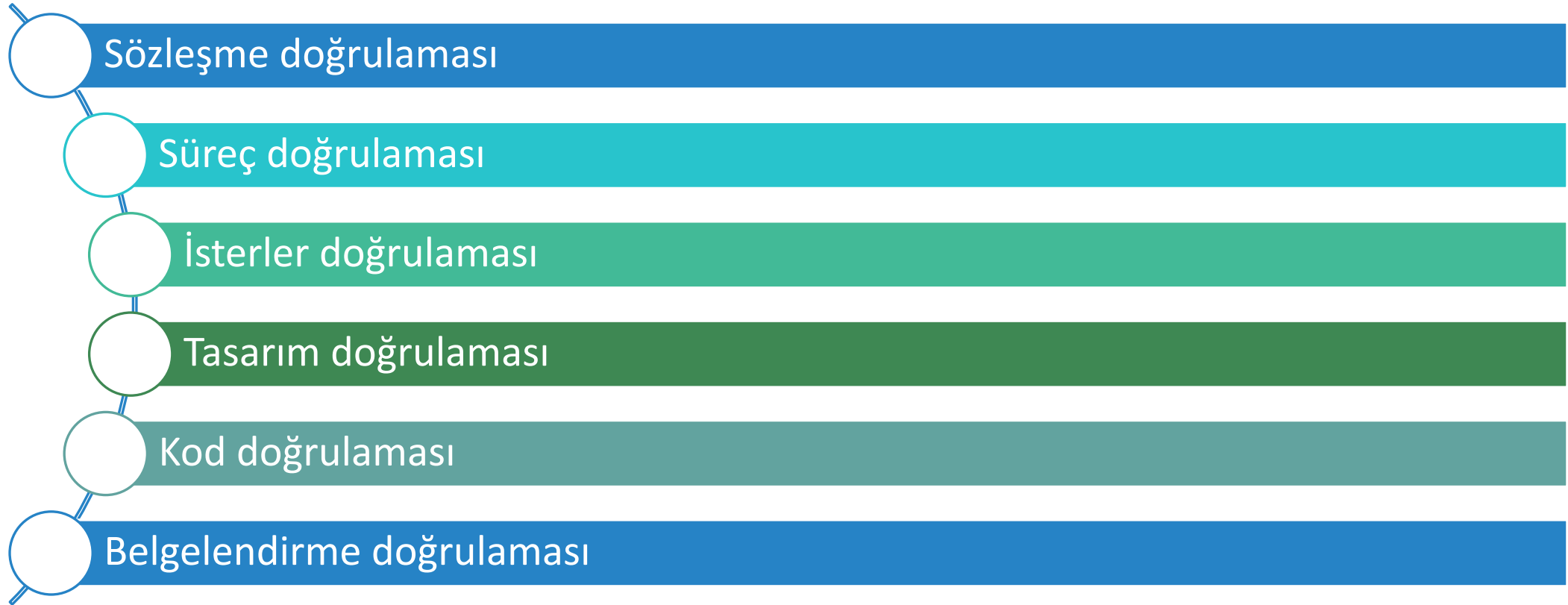
DOĞRULAMA

- Yazılımı doğru mu üretiyoruz?
- Sistemin hatasız ve iyi bir mühendislik ürünü olup olmadığını ölçer.
- Geliştiriciler veya QA ekibi tarafından gerçekleştirilir.
- Doğrulama aşamasında bulunan hataların maliyeti daha azdır.

GEÇERLEME

- Üretilen Yazılım Doğru mu?
- Sistemin kullanıcı gereksinimlerine uygunluğu ölçer.
- Test ekibi tarafından gerçekleştirilir.
- Geçerleme aşamasında bulunan hataların maliyeti daha fazladır.

4.1. Doğrulama Süreci



4.1. Doğrulama Süreci

➤ **Sözleşme Doğrulaması:** Geliştirici ile müşteri arasında yapılan sözleşme şu kısıtlara göre doğrulanmalıdır:



Geliştirici tüm isterleri karşılayabileceği güvenini vermektedir.

İsterler tutarlı olup kullanıcı gereksinimlerini kapsamaktadır.

İsterlere yapılacak değişiklikleri ve ortaya çıkabilecek problemleri kontrol edebilmek üzere yordamlar öngörülmüştür.

Taraflar arasında sahiplik, garanti, telif hakları ve gizlilik gibi konuları da içerecek şekilde işbirliği yapılmaktadır.

İsterlerle birlikte kabul kısıtları ve yordamları öngörülmüştür.

4.1. Doğrulama Süreci

➤ **Süreç Doğrulaması:** Süreç aşağıda listelenen kıstaslara göre doğrulanmalıdır.



Proje planlamaları yeterli ve takvime uygundur.

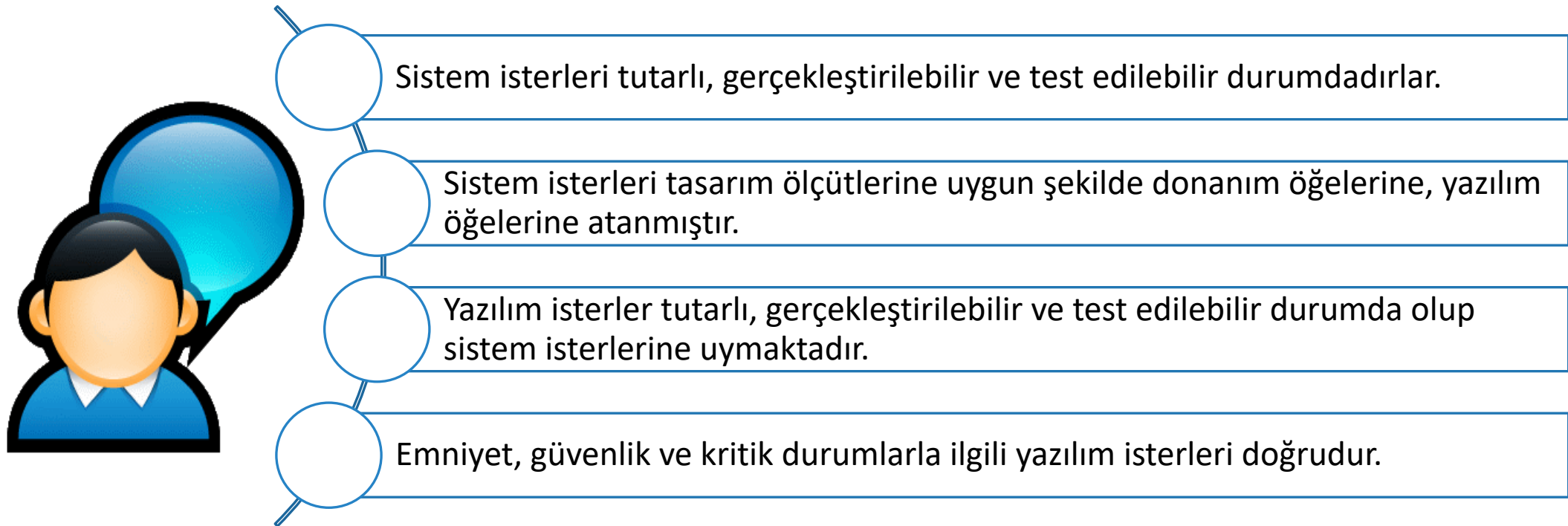
Proje için seçilen süreçler yeterlidir, planlandığı şekilde yürütülmektedir.

Proje süreçleri için seçilmiş standartlar, yordamlar ve ortamlar yeterlidir.

Proje personeli sağlanmış ve sözleşme gereklerine göre eğitilmiştir.

4.1. Doğrulama Süreci

➤ **İsterlerin Doğrulanması:** İsterler şu kriterlere göre doğrulanmalıdır:



4.1. Doğrulama Süreci

➤ **Tasarım Doğrulaması:** Tasarım aşağıdaki kriterlere göre doğrulanmalıdır:



Seçilen tasarım isterlerden türetilmektedir.

Tasarım isterlere göre tutarlıdır ve izlenebilir durumdadır.

Tasarım, olayların, girdilerin, ara yüzlerin, mantık akışının uygun dizilişlerini, zaman ve büyüklük tahsislerini, hata tanımlarını, hataya dayanıklılığını gerçekleştirmektedir.

Tasarım, emniyet, güvenlik ve diğer kritik durumlarla ilgili isterleri gerçekleştirmektedir.

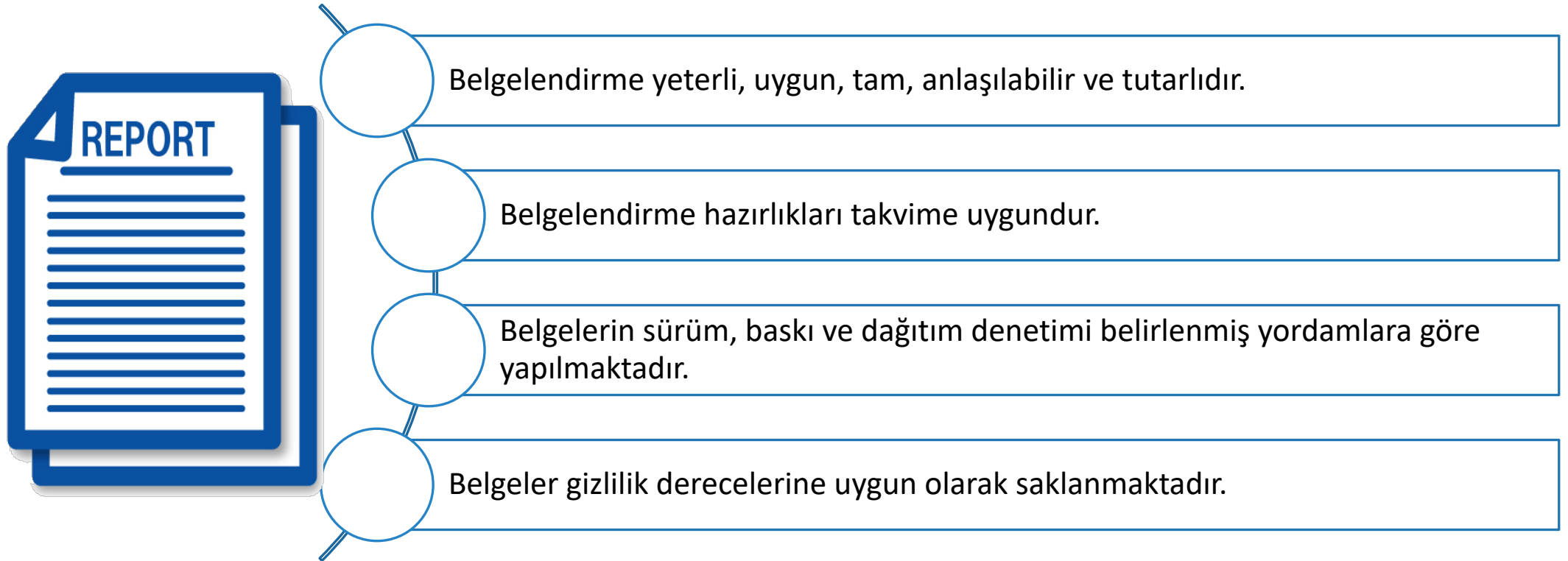
4.1. Doğrulama Süreci

Kod Doğrulaması

- Kod, tasarıma ve isterlere göre izlenebilir, türetilebilir ve test edilebilir.
- Kod doğru ve kodlama standartlarına uygun olmalıdır.
- Kod, olayların, girdilerin, ara yüzlerin, mantık akışının uygun dizilişlerini, zaman ve büyüklük tahsislerini, hata tanımlarını, hataya dayanıklılığı ve geri kazanmayı gerçekleştirmektedir.
- Kod, emniyet, güvenlik ve diğer kritik durumlarla ilgili isterleri gerçekleştirmektedir.
- Kod, yazan kişiden başkası tarafından rahatça okunabilir, anlaşılabilir ve bakım uygulanabilir.

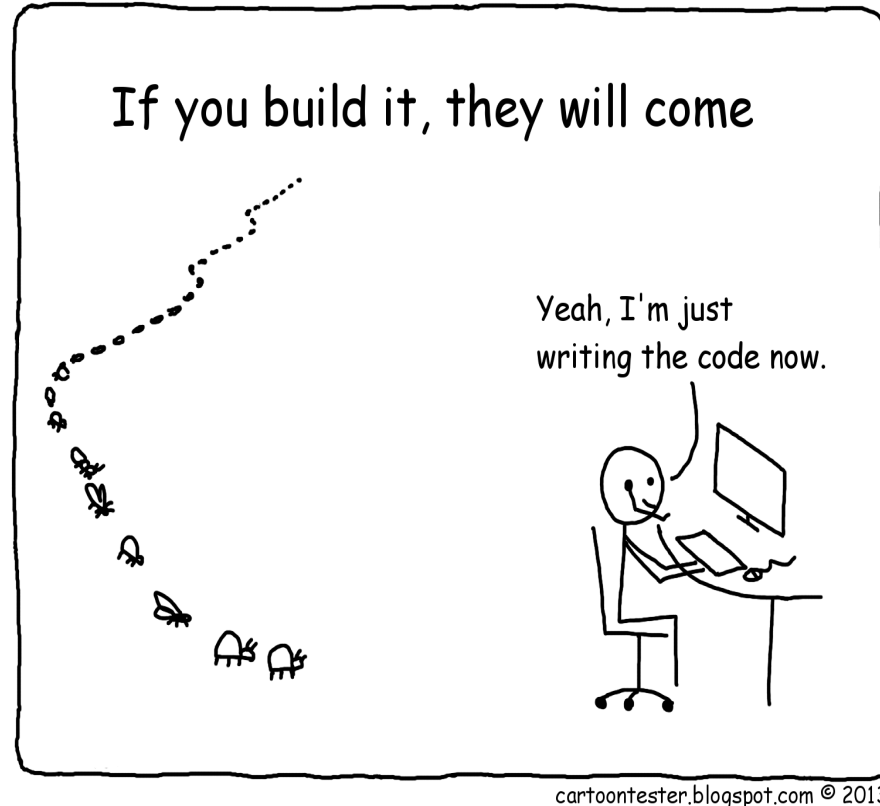
4.1. Doğrulama Süreci

➤ **Belgelendirme Doğrulaması:** Belgelendirme şu kriterlere göre doğrulanmalıdır.



5. Yazılım Hataları

- Yazılım geliştirme süreci boyunca hatalardan kaçınmak imkansızdır.





5. Yazılım Hataları

- Yazılım kalitesi konusunu anlamamanın anahtarı failure, fault ve error terimlerini anlamaktır.
- IEEE Standart 610.12 (IEEE,1990) bu terimleri şu şekilde tanımlamıştır:



5. Yazılım Hataları

- **Error:** Kodlayıcı kaynaklı, doğru olmayan bir sonuç elde edilmesi.
- **Failure:** Sistemin veya bir parçasının gerekli fonksiyonu yeterli performansta yerine getirememesi.
- **Fault:** Bir yazılım içerisindeki doğru olmayan adım, işlem veya veri tanımı.
- Buna göre, failure kullanıcı gereksinimlerini veya ürün özelliklerinin doğru olarak karşılanamaması, fault yazılımdan kaynaklanan hata ve problemler, error ise kodlayıcı kaynaklı problemlerin ortaya çıkması durumu olarak özetlenebilir.

5. Yazılım Hataları

Requirements:

Calculate the hypotenuse of a right-angled triangle.

I.e., for inputs a and b, produce output $\sqrt{a^2 + b^2}$

Implementation:

```
a = readFloat();  
b = readFloat();  
tmp = pow(a,2) * pow(b,2);  
c = sqrt(tmp);  
printFloat(c);
```

User interaction:

input: 3

input: 4

failure
output: 12

Execution:

a = 3

b = 4

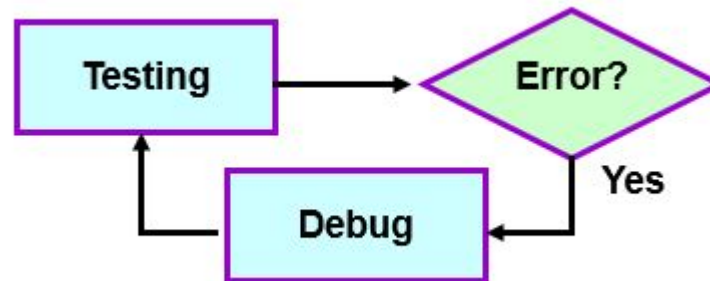
tmp = 144

c = 12

Error
Propagation

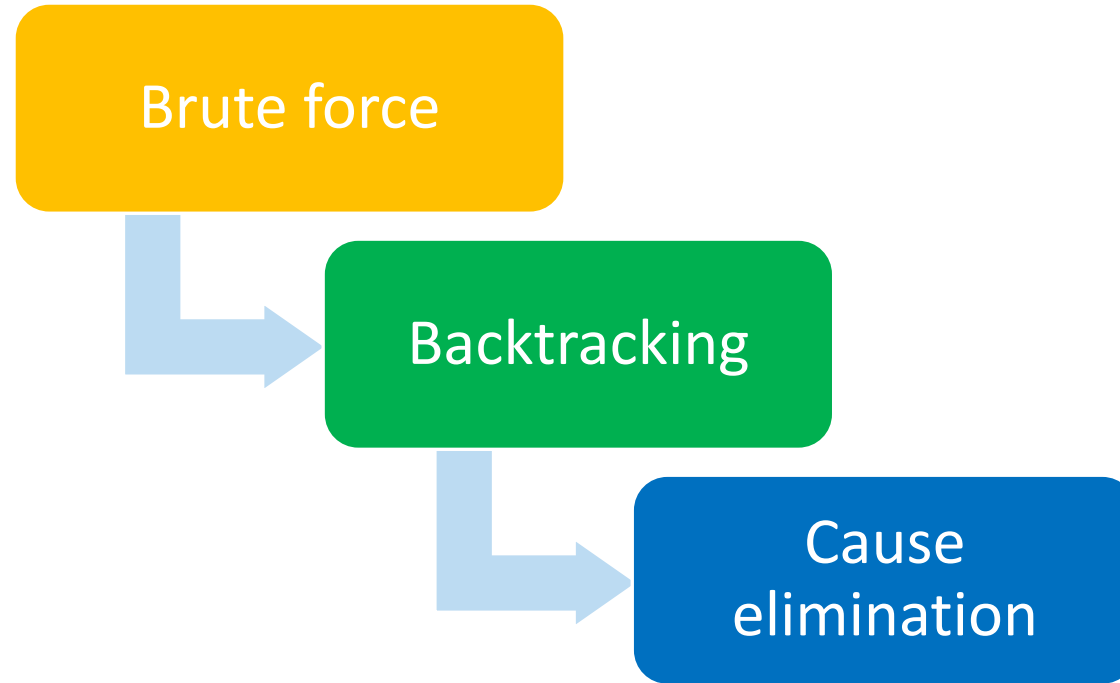
5.1. Hata Ayıklama (Debugging)

- Hata ayıklama başarılı bir test sürecinin sonucudur. Şöyle ki, bir test sonucunda hatalar bulunduğu zaman hataların giderilme işlemine hata ayıklama(debugging) denir.



5. Hata Ayıklama

➤ 3 çeşit hata giderme stratejisi vardır.

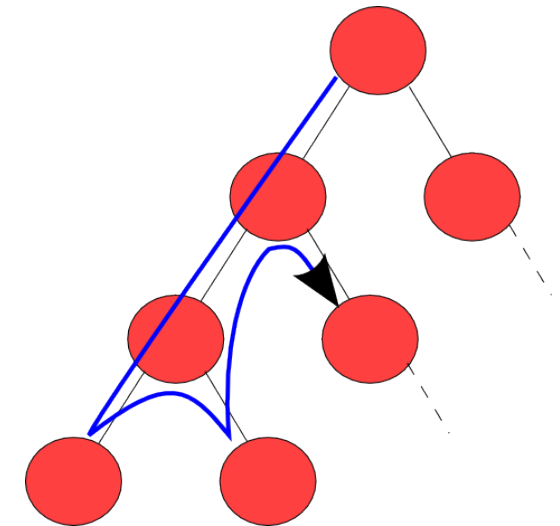


5.1.1. Brute Force

- **Brute Force:** Yürütme anındaki davranışlar izlenir, yazılım biriminin çeşitli noktalarına ekrana veya bir dosyaya o an akışın neresinde olduğunu, genel durumunu veya bir değişkenin değerini yazan deyimler eklenir. Bu bilgiler ışığında, hataya neden olan yazılım kusuru aranır.

5.1.2. Backtracking

➤ **Backtracking:** Kodun okunarak geri izlenmesi esasına dayanır. Hatanın olduğu yerden itibaren geriye doğru gidilerek kod incelenir; hata yaratan deyim yada kusurlu akış mantığı aranır. Küçük yazılımlarda yaygın olarak kullanılır. Yüksek satır ve modül sayısına sahip yazılımlarda, akış yolunun çok dallandığı noktalarda bunu yapmak oldukça zordur.



5.1.3. Cause Elimination

➤ **Cause elimination:** Tümevarım veya tümdengelim yöntemlerine dayanarak elde edilen verilere göre hatanın nedeni araştırılır. Ortaya konan varsayımları doğrulayıcı ya da çürütücü testler tasarlanır. İlk testler olumlu sonuç verirse, daha ayrıntılı verilerle testlere devam edilerek hatanın tam yerinin saptanmasına çalışılır.



6. Testi Kim Yapar?

- Testi kimin yapacağı, test işlemine ve projenin paydaşlarına bağlıdır. Bilişim Teknolojileri sektöründe, büyük şirketler geliştirilen yazılımın yeterliliğini ölçmekle sorumlu test ekiplerine sahiptir. **Yazılım test ekibi**, test edilecek uygulamanın büyüklüğüne ve karmaşıklığına göre oluşturulur.



6. Testi Kim Yapar?

➤ Çoğu durumda aşağıdaki çalışanlar sistemin test edilmesine kendi yaptıkları işle ilgili olarak dahil olurlar:

- Yazılım test ekibi
- Yazılım geliştirici
- Proje lideri
- Son kullanıcı



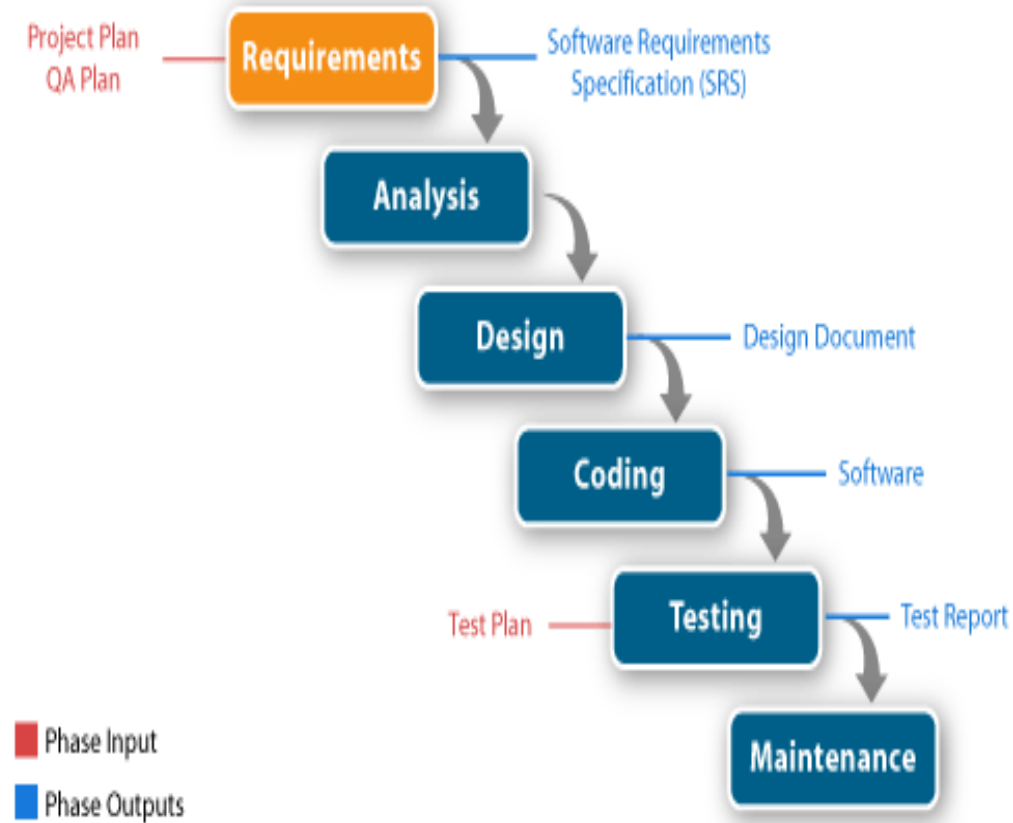
6. Testi Kim Yapar?

- Test Sürecinde mümkün olduğu kadar test ekibi daha etkin kullanılmalıdır ve diğer pozisyonların test sürecindeki etkinliği azaltılmalıdır.
- Test yapan kişilerin,
 - Tedbirli,
 - Meraklı,
 - Eleştirmen
 - İyi iletişim kurabilen kişilikte olmaları gerekir.

7. Yazılım Geliştirme Metodolojilerinde Test

Ş
e
l
a
l
e

M
o
d
e
l
i



7. Yazılım Geliştirme Metodolojilerinde Test

- **Şelale Modeli:** Yazılım süreci lineardir, bir sonraki safhaya geçebilmek için bir önceki safhada yer alan aktivitelerin tamamlanmış olması gerekir. **Yani test aşamasına gelebilmek için diğer aşamalar tamamlanmalıdır.** Hatalar sadece 5. aşamada giderilebildiğinden yazılımın maliyetini artırır ve başarısını azaltır.
- Kullanıcı katılımı başlangıç safhasında mümkündür. Kullanıcı isterleri bu safhada tespit edilir ve detaylandırılır. Daha sonra gelen tasarım ve kodlama safhalarında müşteri ve kullanıcılar ile diyaloga girilmez. Bu yüzden kullanıcı testlerinde başarı yakalama oranları azdır.

7. Yazılım Geliştirme Metodolojilerinde Test

V

M

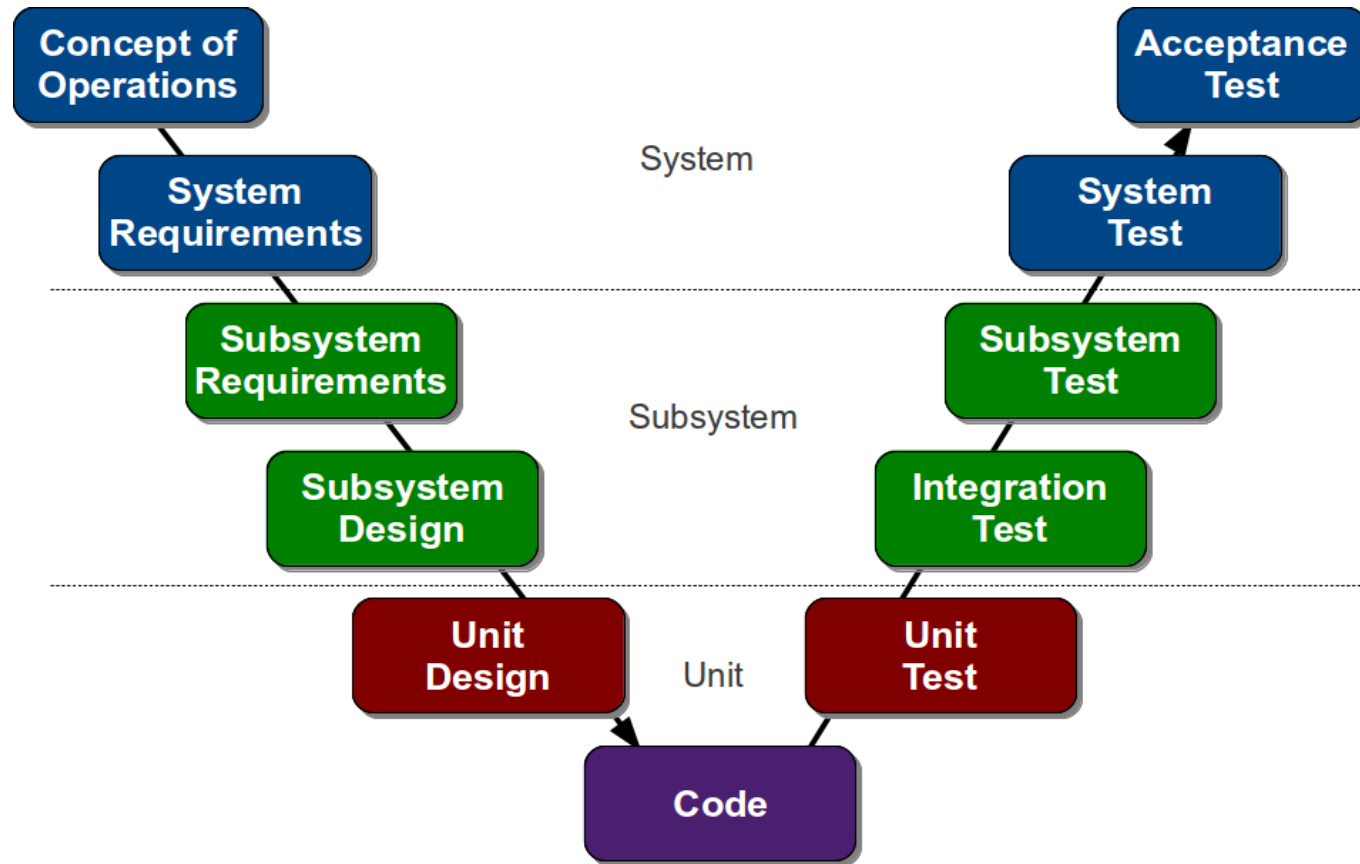
o

d

e

i

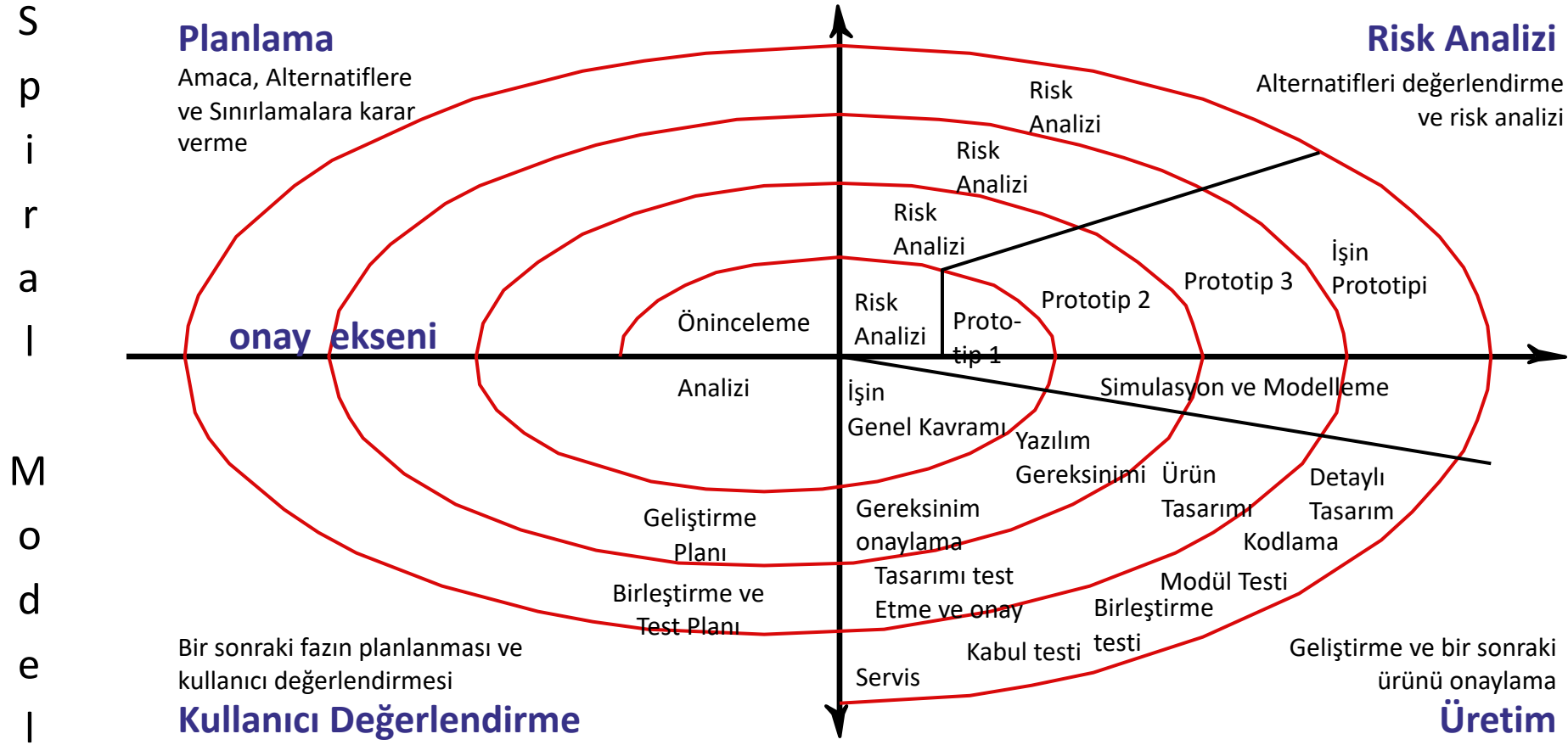
i



7. Yazılım Geliştirme Metodolojilerinde Test

- **V modeli:** Test işlemlerinin ne zaman yapılacağını ön plana çıkarır. Bir önceki şekle bakacak olursak, sol kanat üretim etkinliklerini, sağ kanat da test etkinliklerini gösterir. **Bu modelde geliştirme ve test paralel şekilde yapılır.** Her aşama sonunda test edilecek ürün, test grubu tarafından sınanır, onay verildikçe bir sonraki aşamaya geçilir.
- İsterlerin iyi tanımlandığı, belirsizliklerin az olduğu ve aşamalar halinde ilerlenmesi gereken projelerde «v» modeli iyi sonuç verir.

7. Yazılım Geliştirme Metodolojilerinde Test



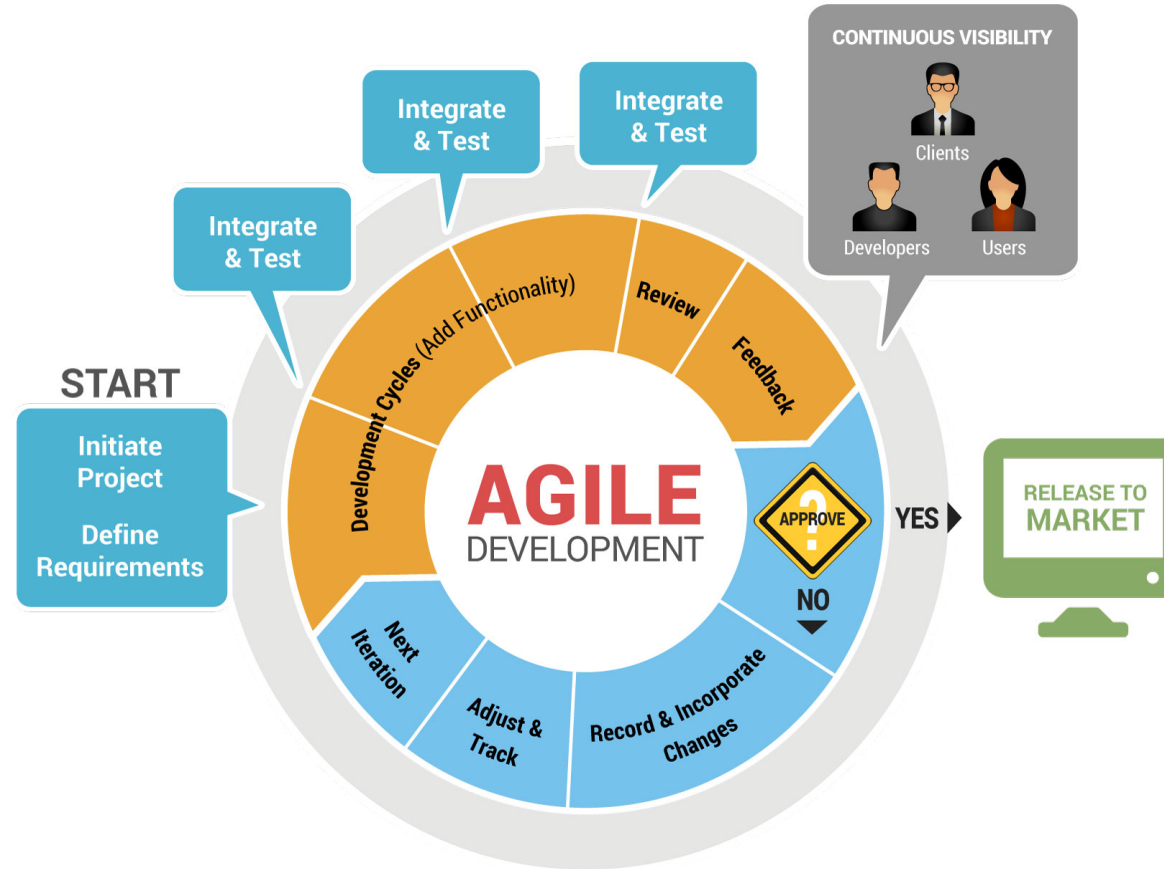
7. Yazılım Geliştirme Metodolojilerinde Test

- **Spiral Model:** Aynı safhalara geri dönülmesinin bir zorunluluk olduğunu vurgular. **Proje çevrimlere ayrılır ve her bir çevrimin riskleri ayrı ayrı ele alınır.**
- Bu modelin avantajları vardır. İlki üretim süreci boyunca ara ürün üretme ve üretilen ara ürünün kullanıcı tarafından sınanması temeline dayanır. Diğer bir avantajı gerek proje sahibi, gerekse yüklenici tarafındaki yöneticiler, çalışan yazılımlarla proje boyunca karşılaştıkları için daha kolay izleme planlaması yapılır. Avantajlarından biri de; yazılımın kodlanması ve testi daha erken başlar.

7. Yazılım Geliştirme Metodolojilerinde Test

Ç
e
v
i
k

Y
a
z
ı
l
ı
m



7. Yazılım Geliştirme Metodolojilerinde Test

➤ **Agile:** Geleneksel yaklaşımın tersine, **test profesyonelleri, yazılım geliştirme yaşam döngüsünün en başından itibaren sürece dahil olurlar.** Kullanıcının geliştirici ve test sorumlusu ile aynı ortamı paylaşması prensibi, geliştirilen ürünün kullanıcının gerçekten istediği bir biçimde geliştirilmesi ve gerçekçi kullanım durumlarına göre test edilmesini sağlar. Test güdümlü geliştirme yöntemini(TDD) kullanır.

TDD: Test Driven Development yöntemiyle kodlama yaparken genelde şu şekilde bir yol izlenmektedir.

1. Tek satır kod yazmadan kodun testini yaz.
2. Testi çalıştır ve testin geçemediğini gör.
3. Testi geçecek en basit kodu yaz ve testin geçtiğini gör.
4. Kodu düzenle
5. Başa dön

Sorularınız



Çalışma Soruları

1. Yazılım hatalarını örnek vererek açıklayınız.
2. Hangi yazılım geliştirme metodolojisinde testin yeri en büyüktür? Açıklayınız.
3. Hata ayıklama nedir? Test ile farkını yazınız.
4. Hata ayıklamanın türlerini yazıp açıklayınız.
5. Doğrulama ve geçerleme arasındaki farkları yazınız. Birer örnekle açıklayınız.
6. Doğrulama süreci aşamalarını yazınız.
7. Testin amaçları nelerdir? Yazılım testinde neler test edilir?
8. Test güdümlü geliştirme nedir? Avantajları nelerdir?

Kaynaklar

- [1] Software Engineering A Practitioner's Approach (7th Edition), Roger Pressman, 2013
- [2] Yazılım Mühendisliği (2. Baskı), M. Erhan Sarıdoğan, 2008
- [3] Yazılım Test Mühendisliği (1. Baskı), Rifat Çölkesen, 2010
- [4] <http://www.cihatahtuntas.com/test-driven-development/>
- [5] www.kurumsaljava.com/download/10/
- [6] <http://www.turkishtestingboard.org/TurkeySoftwareQualityReport.pdf>
- [7] http://www.tutorialspoint.com/software_testing/