SOFTWARE DEVELOPMENT LIFE CYCLE

(Yazılım Geliştirme Yaşam Döngüsü)

2. Ders
11.06.2022

Aklımızda Kalanlar?

1- SDLC yazılım geliştirme yaşam döngüsü
Amacı : high quality , user expectation karşılayan ürünler oluşturmak

Planlama
Analiz
Tasarım
Kodlama
Test
Teslimat ve Bakım

Aklımızda Kalanlar?

1-Roller

Project manager:

Planlama aşamasında yer alır
Ekibi koordine eder
Stake holder ile bağlantı
Bütçe,

Business Analist(expert- uzman)
FRD , BRD sahibi

Development Team(Developer ve tester)

YAZILIM GELİŞTİRMEDE KULLANILAN MODELLER

1968 yılından itibaren; yazılım geliştirme, uygun, etkili, güvenilir yazılımları mümkün olduğunca az bir maliyetle elde edebilme yolundaki çalışmaların artmasıyla YAZILIM MÜHENDİSLİĞİ olarak adlandırılan disiplin kurulmuştur.

Yazılım Mühendisliği'nin temel hedefi; sistematik ve organize bir yaklaşımı çalışma alanlarına adapte etmek ve bir problemin çözümü için uygun tüm araç ve teknikleri kullanmaktır. Yazılım ürünü ise, bir sistem projesi olarak, bir ekip çalışmasıyla, belli bir sürede hazırlanmaktadır. Yazılım ürünlerinin süresi ve bütçesi; projeye bağlı olarak değişiklik göstermektedir.

YAZILIM GELİŞTİRMEDE KULLANILAN MODELLER

Yazılım geliştirme modeli; yazılımın gerçekleştirilebilmesi için gerekli stratejiyi ifade eder ki bu strateji bir dizi aktiviteyi ve olayları içermektedir. Ancak seçilen her modelin kendine özgü avantaj ve dezavantajları bulunmaktadır. Dikkat edilmesi gereken nokta çözüm istenen ürüne ve sürece uygun modelin seçilebilmesidir.

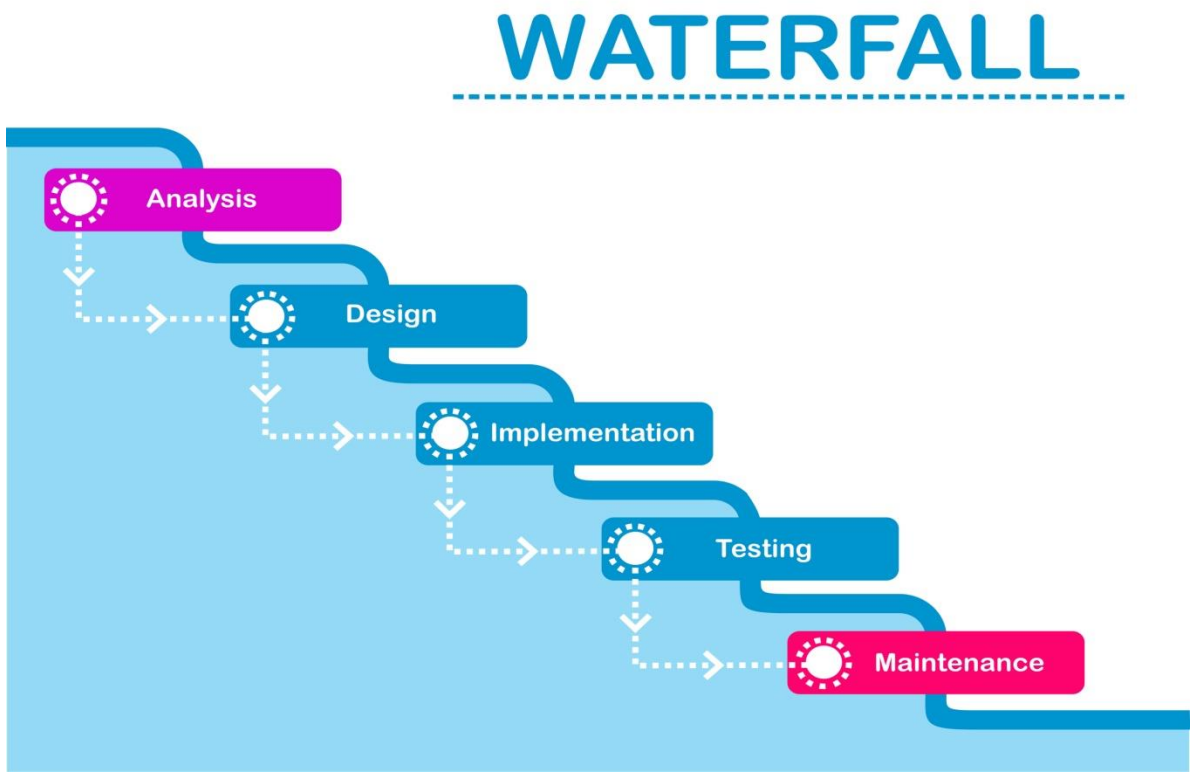
En klasik yazılım geliştirme modeli metodolojisi, **şelale modelidir.** Şelale modeli dışında, **v modeli**, **prototip modelleme**, **spiral model**, **çevik geliştirme** gibi pek çok model bulunmaktadır.

YAZILIM GELİŞTİRMEDE KULLANILAN MODELLER

Yazılım Geliştirme Metodolojilerinde en yaygın kullanılanlar;

1) WATERFALL MODEL(Şelale Modeli)
2) V MODEL
3) SPIRAL MODEL
4) PROTOTİP MODEL
5) AGILE METHODOLGY (Çevik Metodoloji)

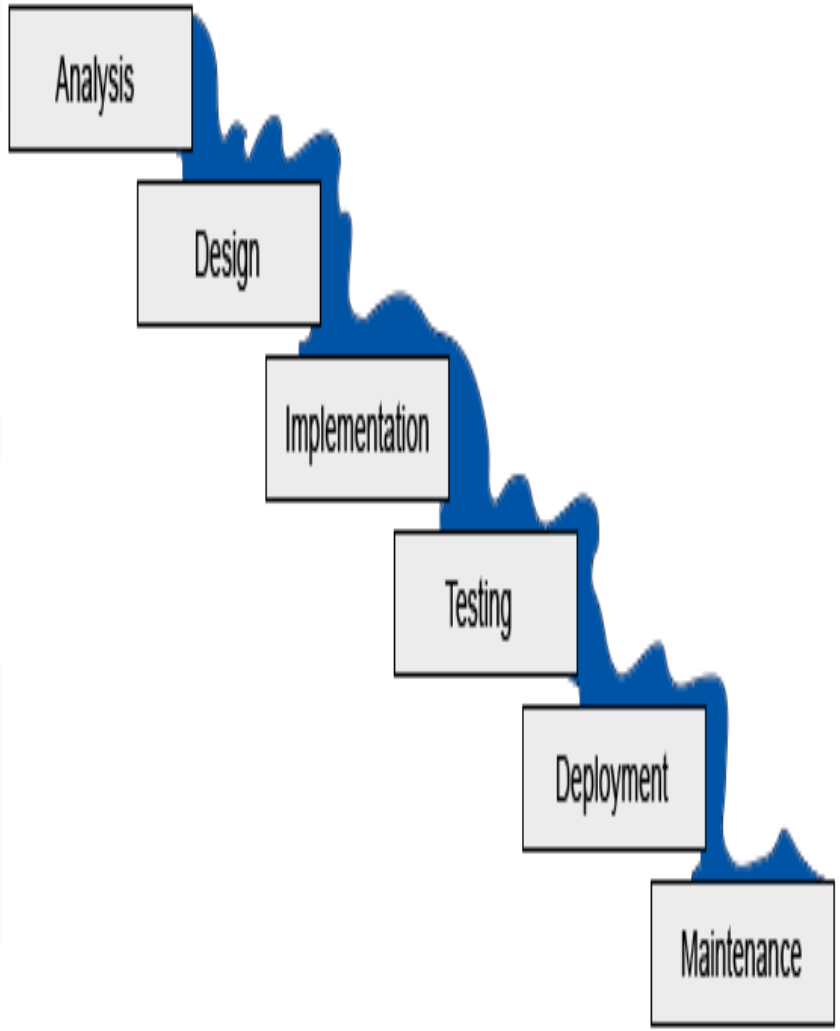
WATERFALL MODEL(Şelale Modeli)



- Şelale modeli (Waterfall) proje yönetim süreci; analiz, tasarım, yazılım, test, yayın gibi fazlardan oluşur.
- Geleneksel bir yöntemdir; süreçler tıpkı bir şelale gibi yukarıdan aşağıya doğrusal olarak işler.
- Bir faz tamamlanıp yenisine geçildiğinde, bir önceki faza geri dönülmez.
- Proje sahibi, proje tamamlandıktan sonra ürünü görebilir.

WATERFALL MODEL(Şelale Modeli)

- Şelale modeli, analiz adımı ile başlar. Analiz adımında, tüm yazılım gereksinimleri net bir şekilde belirlenerek analiz dokümanı üretilir.
- Daha sonra, tasarım adımında; yazılımın arayüz, veritabanı, sınıf vb. tasarımları yapılarak tasarım dökümanı üretilir.
- Bir sonraki kodlama adımında yazılım; analiz ve tasarım dokümanlarında belirtilen şekilde kodlanır.
- Test adımında; analiz ve tasarım dokümanlarındaki tüm fonksiyonel ve fonksiyonel olmayan gereksinimler ve tasarımlar için test senaryoları yazılır ve bu test senaryoları icra edilerek yazılımın testleri yapılır.
- Test adımı sonunda, yazılımda herhangi bir hatası bulunamaz ise, entegrasyon adımına geçilir ve yazılım, canlı ortama entegre edilerek müşterinin kullanımına açılır.



WATERFALL MODEL(Şelale Modeli)

AVANTAJLARI

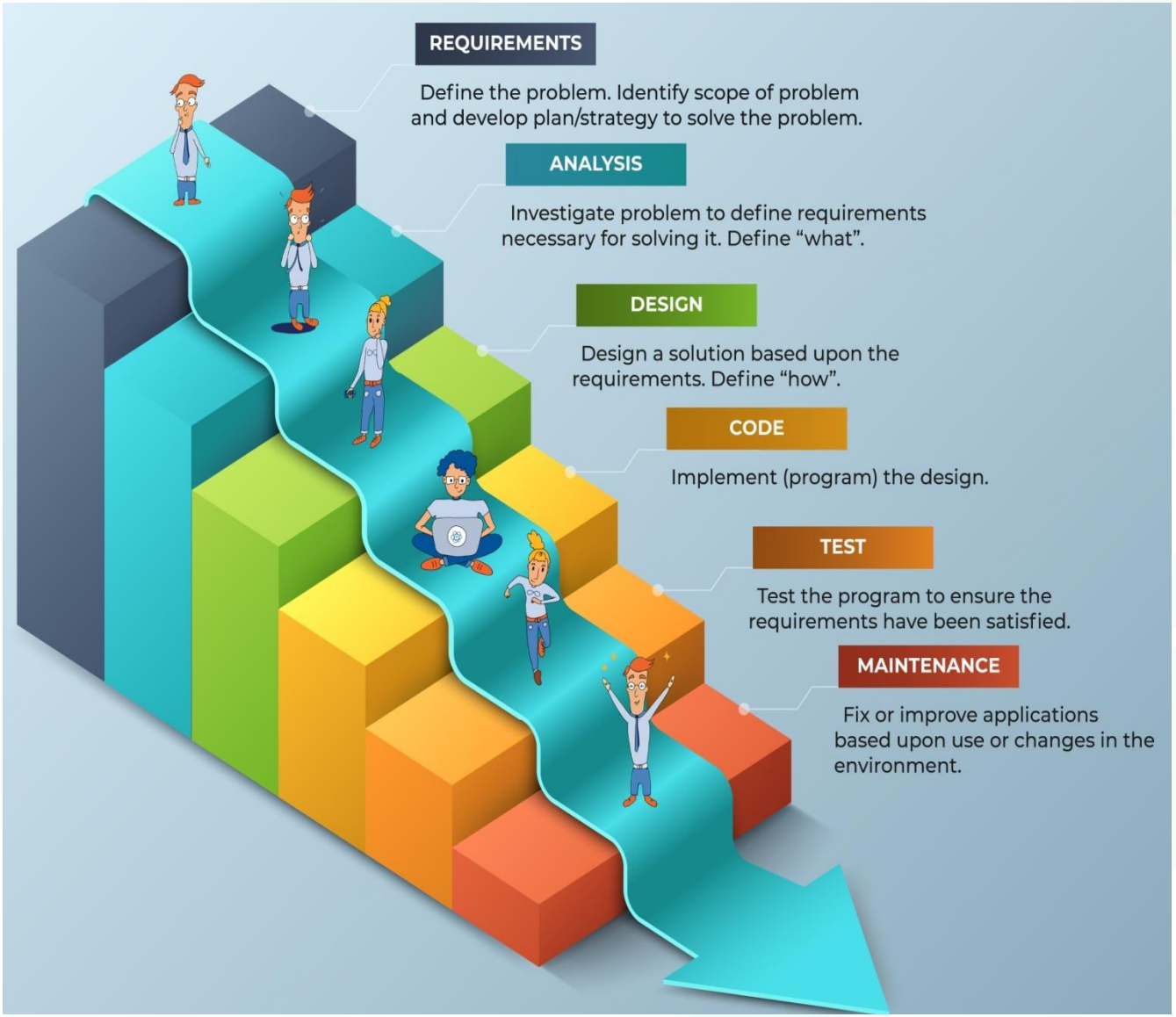
- Kullanımı ve yönetimi kolaydır.
- Gereksinimler iyi anlaşılır.
- Proje bilgisini aktarmak daha kolaydır.
- Küçük projeler için daha iyidir
- Görevler mümkün olduğunca sabit kalır
- Kapsamlı dökümanlar oluşturulur.

DEZAVANTAJLARI

- Değişim ve yenilik zordur.
- Müşteri öngörü ve önerileri önemsenmez.
- Projenin bitimine kadar çalışan ürün yok.
- Beklenmedik riskleri kolayca ele alamıyor.

WATERFALL MODEL(Şelale Modeli)

Şelale modelinde analiz ve tasarım aşamaları oldukça detaylı yapıldığından, bu adımlar uzun sürmektedir. Ancak, analiz ve tasarım aşamalarında gereksinimlerin ve tasarımın net bir şekilde ortaya konulmasından dolayı, kodlama ve test aşamaları çok kısa sürmektedir. Test aşamasında çıkan hata sayısı azdır.

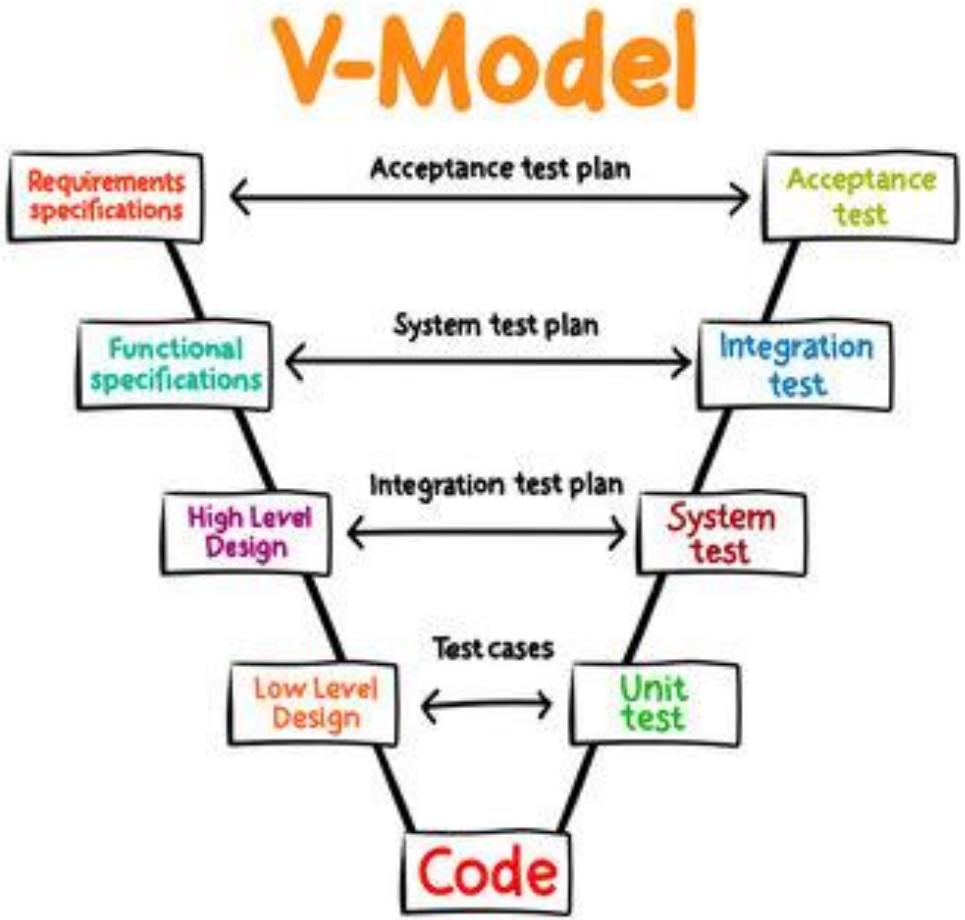


WATERFALL MODEL(Şelale Modeli)

- Şelale modelinde, üst adımlarda yapılan hataların yarattığı zaman kaybı oldukça fazladır. Örneğin test aşamasında karşılaşılan bir hatanın analizden kaynaklandığı tespit edilirse, analiz, tasarım ve test dokümanlarının güncellenmesi ve kodun düzeltilmesi gerekir, ki bu oldukça ciddi zaman ve dolayısıyla para kaybına yol açar.
- Şelale modelinin dezavantajlarından bir tanesi de, ürünün ortaya çıkması için tüm aşamaların tamamlanmasını beklemek zorunda kalmaktır. Örneğin; proje 4 sene sürecek ise, müşterinin ilk prototipi görmesi için, analiz, tasarım ve kodlama aşamalarının bitmesini, yani en az 2-3 sene beklemesi gerekecektir. Bu durum; bazı sabırsız müşteriler için sorun teşkil edebilir. Bu gibi durumlarda Agile yöntemler daha faydalı olabilir.

V MODEL

V modeli, **Doğrulama(verification) ve Geçerleme (Onaylama-validation)** modeli anlamına gelir. Tıpkı şelale modelinde olduğu gibi yazılım yaşam döngüsü adımları V-şeklinde sıralı bir şekilde uygulanır. Bu modelde de her aşama bir sonraki aşama başlamadan önce tamamlanmalıdır. V-Modelinin en temel özelliği “ürünün test edilmesi kendisine karşılık gelen geliştirme aşamasına paralel olarak planlanmaktadır.”



V MODEL

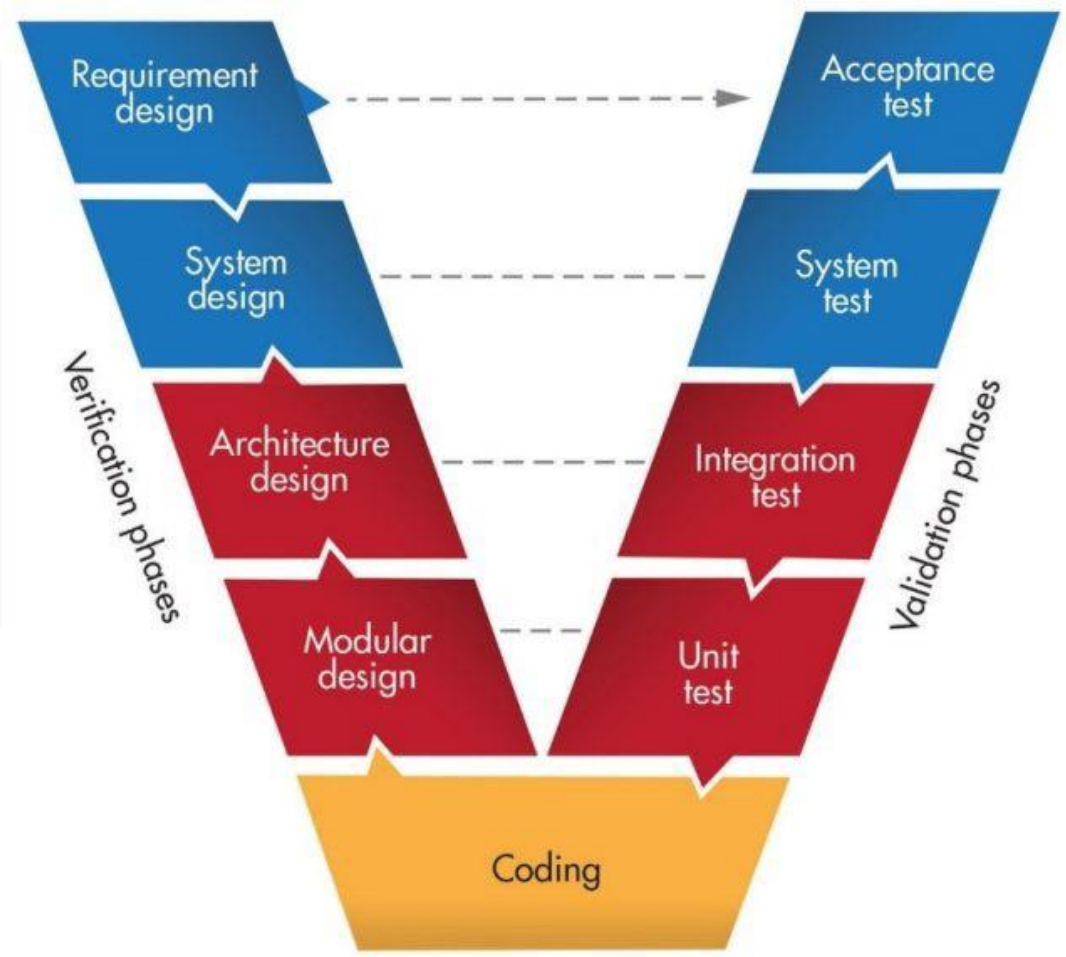
V-modelinin avantajları:

•Basit ve kullanımı kolaydır.

•Planlama ve test tasarımı gibi test faaliyetleri kodlamadan önce gerçekleştirildiği için proje içerisinde çok zaman kazandırır. Bu nedenle şelale modeline göre daha yüksek başarı şansı vardır.

•Hataların bulunması erken aşamada bulunur.

•Hataların bir sonraki aşamaya geçmesi önlenir.



V MODEL

V modelinin dezavantajları:

•Uygulama şekli oldukça katı, kesin kurallara bağlıdır.

•Yazılım şelalede olduğu gibi geliştirme aşamasında geliştirilir, bu nedenle yazılımın erken prototipleri üretilmez.

•Herhangi bir aşamada gereksinimler üzerinde değişiklik olursa, test belgelerinin de diğer belgeler ile birlikte güncellenmelidir.

V modeli, gereksinimlerin açıkça tanımlandığı projeler için kullanılabilir.

SPIRAL MODEL

Tasarımı doğrusal bir süreç olarak gören diğer modellerin aksine, bu model spiral bir süreç olarak görür. Bu, yineleyici tasarım döngülerini genişleyen bir spiral olarak temsil ederek yapılır. Genellikle iç çevrimler, gereksinim tanımının rafine edilmesi için prototipleme ile birlikte ihtiyaç analizinin erken evresini ve dış spiraller yazılım tasarımını aşamalı olarak temsil eder.
Her helezonda, tasarım çabalarını ve bu yineleme için ilgili riski değerlendirmek için bir risk değerlendirme aşaması vardır. Her spiralin sonunda, mevcut spiralin gözden geçirilebilmesi ve bir sonraki aşamanın planlanabilmesi için gözden geçirme aşaması vardır.

Spiral Process Model

SPIRAL MODEL

Her tasarım sarmalının altı ana faaliyeti altı temel görevle temsil edilmektedir:

1. Müşteri İletişimi
2. Planlama
3. Risk Analizi
4. Yazılım Tasarımı
5. Üretim-dağıtım
6. Müşteri onayı

Avantajları

1. Risk analizi yapmaktadır.
2. Bu yazılım tasarım modeli, büyük yazılım projelerini tasarlamak ve yönetmek için daha uygundur.

Dezavantajları

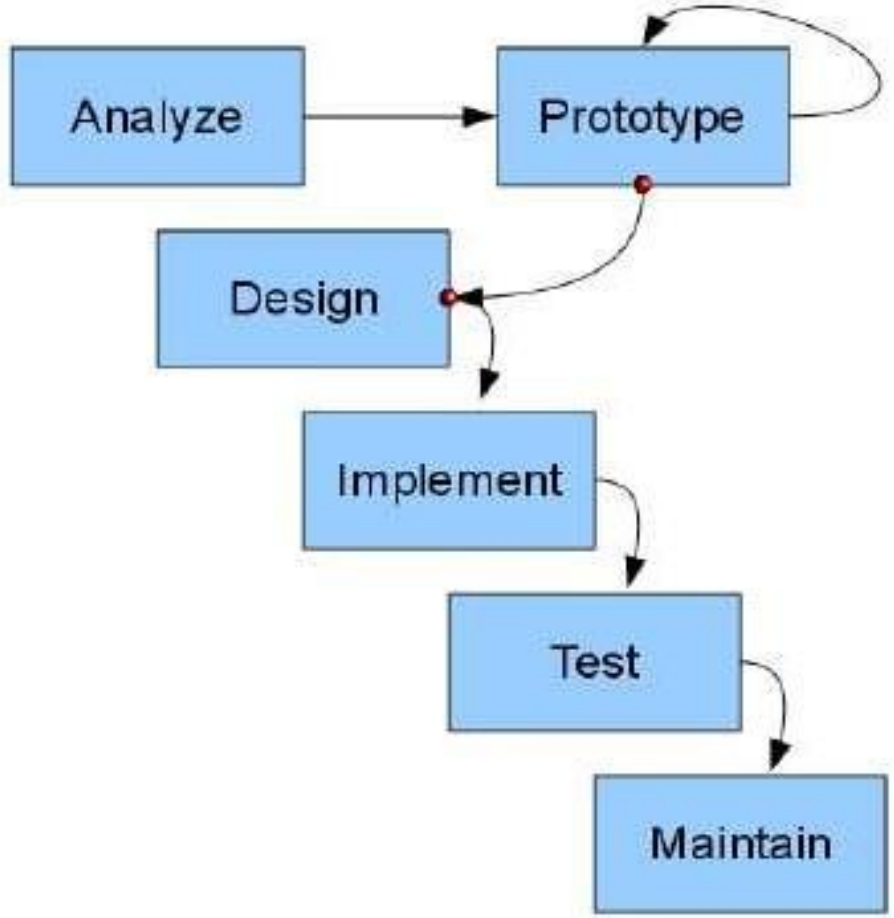
1. Risk analizi yüksek uzmanlık gerektirir.
2. Kullanması pahalı model
3. Küçük projeler için uygun değildir.

Spiral Process Model

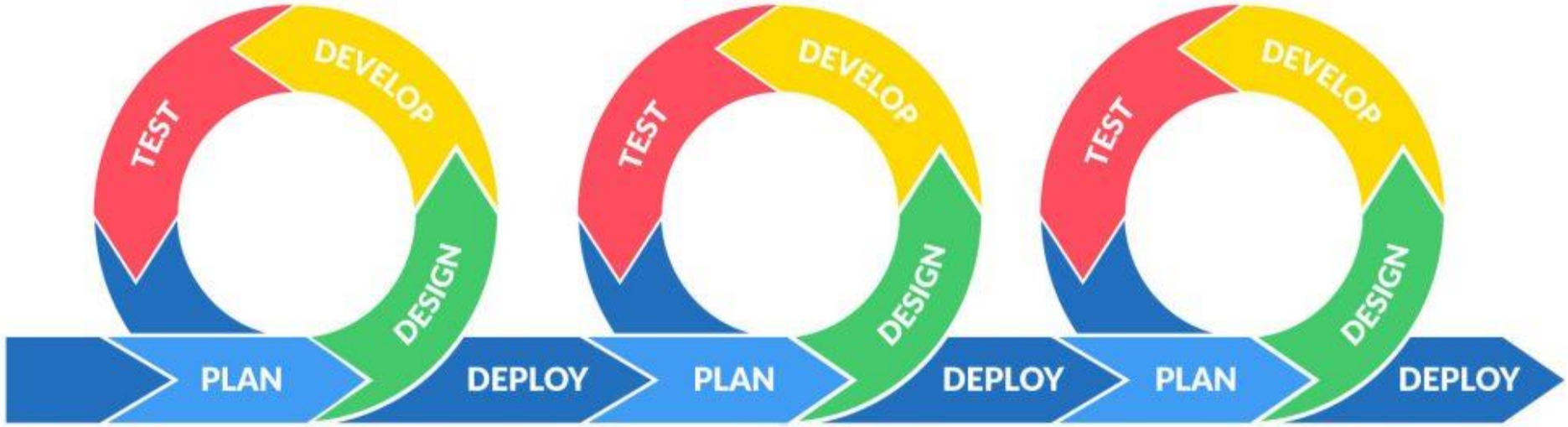
PROTOTİP MODEL

- Bu modelde çabuk tasarım, prototip geliştirme ve müşteri değerlendirilmesinden sonra prototip iyileştirilip referans ürün ortaya konur.

- Müşteriye sunulan ön ürün; ilk ürün olarak kabul edilir, yada iptal edilip en baştan yapılabilir.



AGILE METHODOLGY (Çevik Metodoloji)



Agile Metodoloji (Çevik Metodoloji) yazılım sistemlerini etkili ve verimli bir şekilde modellemeye ve dokümantasyonunu yapmaya yönelik, pratiğe dayalı bir yöntemdir.

AGILE METHODOLGY (Çevik Metodoloji)

Aşırı kuralcı klasik yazılım süreç modellerine tepki olarak ortaya çıkmıştır. Yazılımlar daha yüksek maliyetli ve daha yavaş geliştirilmekteydi. Yazılım geliştirme sürecini hızlandırmak, daha etkin kullanmak ve gerektiğinde dökümante etmek amacıyla bir çok yaklaşım ortaya çıkmıştır.

- 2001 yılında yazılım dünyasının önde gelen isimlerinden 17 arkadaş; "**Agile (Çevik) Yazılım Geliştirme Manifestosu**" ve "Agile (Çevik) Yazılımın Prensipleri" ni yayınlamışlar, bu oluşumu ve gelişimini desteklemek için "Agile Alliance" adıyla, kar amacı gütmeyen bir organizasyon kurmuşlardır.
- Manifesto, nasıl daha iyi bir yazılım geliştirdiklerini ve bunu yapmak isteyenlere yol gösterecek 12 maddeden oluşmaktadır.

AGILE METHODOLGY (Çevik Metodoloji)

AGILE MANIFESTO

1) İlk önceliğimiz kaliteli yazılımı müşteriye teslim edebilmektir. Bu projenin ilk aşamalarından itibaren sürekli teslimlerle yapılır ve müşterinin yazılımı çok önceden kullanmaya başlayarak deger sağlamasına olanak sağlanır. Günümüzde çevik süreçlere artan ilginin başlıca nedenlerden biri , yapılan yatırımların hızlı geri dönüşünün olmasıdır.

2) Değişiklikler projenin ileriki aşamalarında dahi olsa kabul edilir. Amaç müşterinin ihtiyaçlarını karşılayan,onlara yarar sağlayacak, gerçek değer katacak yazılım üretmektir ve ihtiyaçlarda meydana gelen değişiklikler projenin sonraki aşamalarında dahi yazılıma aksettirilmelidir. Test, güdümlü tasarım, kapsamlı otomatik testler, sürekli entegrasyon, basit tasarım gibi pratikler sayesinde değişikliklerin getireceği maliyetler minimuma indirilir ve süreç değişikliklere çabuk adapte hale getirilir.

AGILE METHODOLGY (Çevik Metodoloji)

AGILE MANIFESTO

3) Çok kısa aralıklarla yazılım teslimleri yapılır. Bu aralıklar tipik olarak 2-4 hafta arasıdır. Bu sayede sürekli geri beslenim (feedback) sağlanır ve müşterinin tam istediği şekilde yazılım geliştirilerek ilerler.

4) Alan uzmanları , yazılımcılar, testçiler günlük olarak birlikte çalışırlar. Farklı roller arasında duvarlar örülmez. Rol bazlı ekipler yerine yazılım özelliklerine(features) göre ekipler oluşturulur. Analist (BA), yazılım geliştirici(Dev), tester (QA) vb. aynı ekibin içinde çalışır ve sürekli iletişim halindedir.

5) Motive olmuş bireyler etrafında projeler oluşturun. Ekip üyelerine kendileri ile ilgili alacakları kararlar konusunda güvenilir. Ekip kendi kendine organize olacak yetkiye sahiptir. Onlara ihtiyaç duydukları ortamı ve desteği verin ve işi tamamlamaları için onlara güvenin.

AGILE METHODOLGY (Çevik Metodoloji)

AGILE MANIFESTO

6) Bir geliştirme ekibine ve içinde bilgi aktarmanın en verimli ve etkili yöntemi yüz yüze görüşmedir.

7) Çalışan yazılım, ilerlemenin birincil ölçüsüdür. Projedeki gelişmenin tek ölçüsü o ana kadar geliştirilmiş özellikler ve çalısan yazılımdır.

8) Agile processes (süreçler) sürdürülebilir gelişmeyi teşvik eder. Planlamaların sağlıklı olması için ekibin iş teslim hızının üzerinde çok oynanaması gerekir. Örneğin fazla mesailer gibi yöntemlerle ekibin hızını geçiçi olarak arttırmak tercih edilen yöntemlerden değildir.

9) Teknik mükemmelliğe ve iyi tasarıma verilen sürekli dikkat, çevikliği artırır.

AGILE METHODOLGY (Çevik Metodoloji)

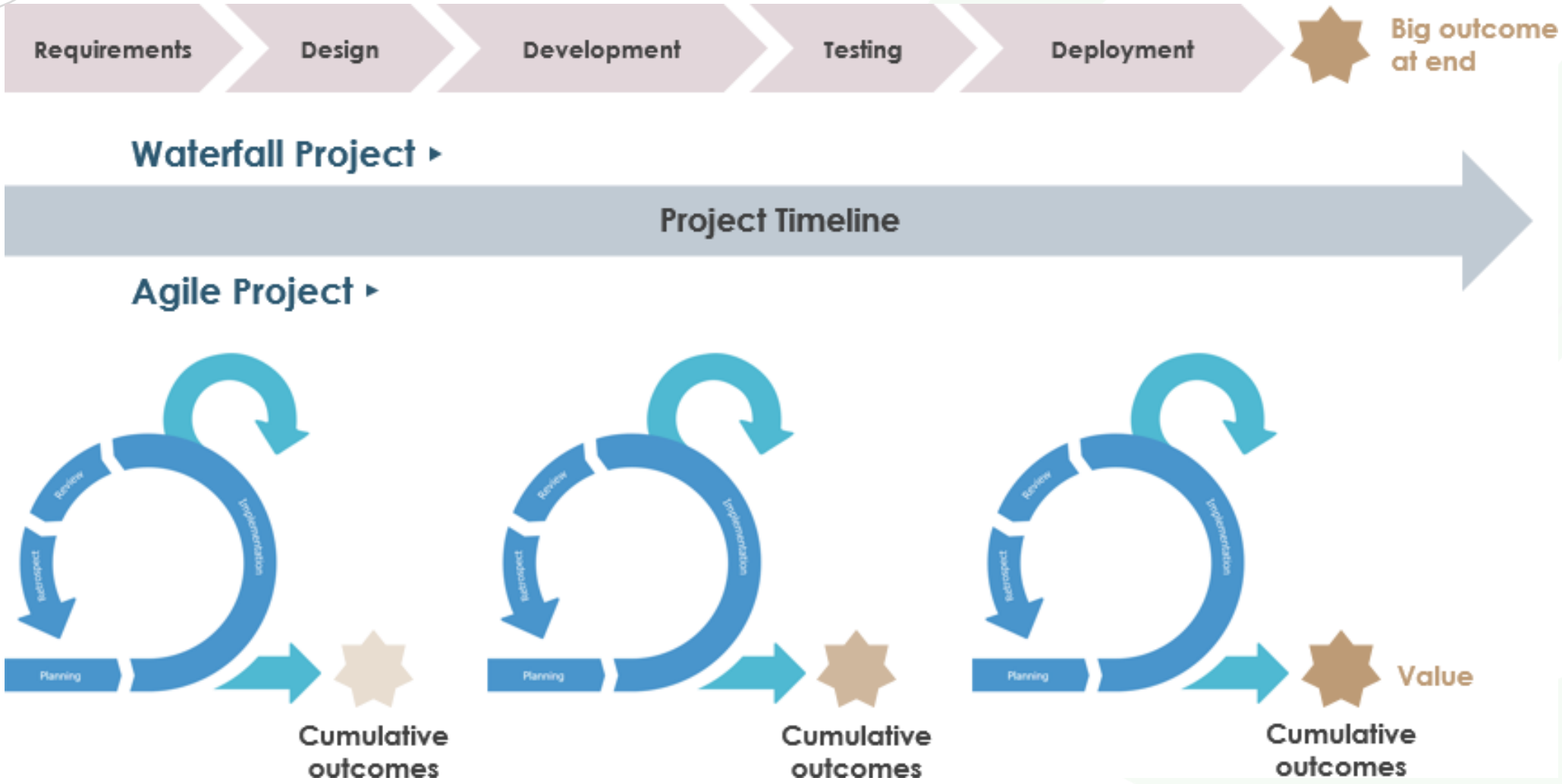
AGILE MANIFESTO

10) Sadelik esastır. Sadelik anlayışı akla gelen baştan savma çözümü uygulamak yerine, anlaşılması ve sonradan değiştirilmesi kolay , maliyeti en düşük ve o an ki gereksinimleri karşılayan çözümü kullanmaktır.

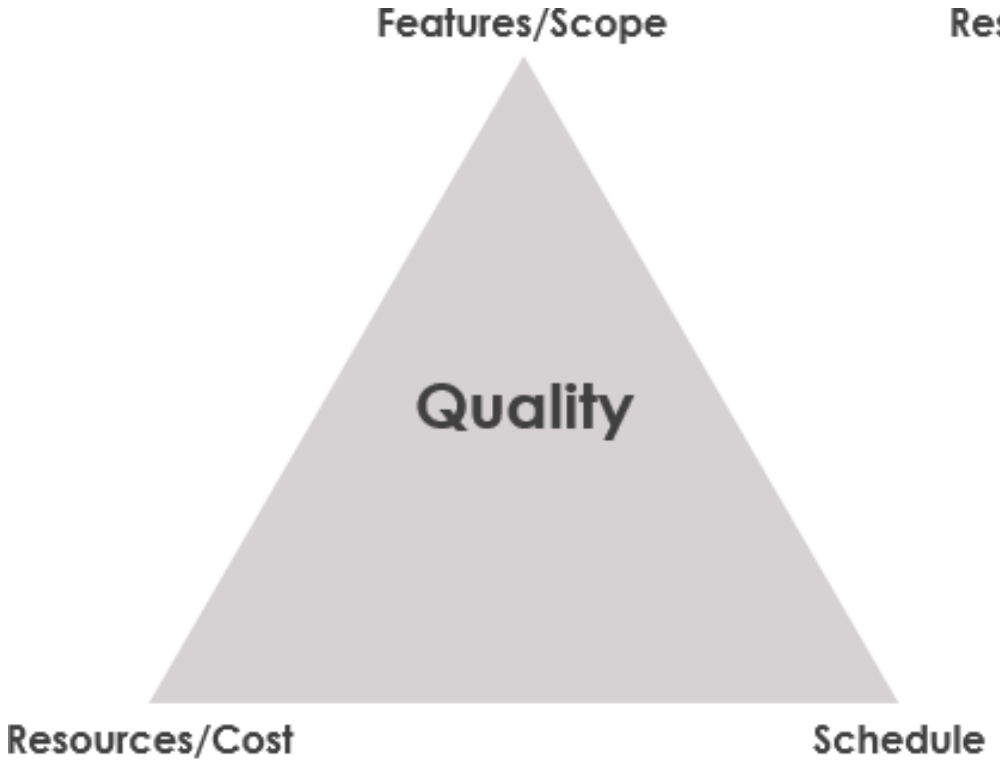
11) En iyi mimariler, gereksinimler ve tasarımlar kendi kendini organize eden ekiplerden ortaya çıkar. En etkin çalışan ekipler kendilerini organize edebilen , bu konuda yetkin ekiplerdir. Ekip kendi çalışma yöntemlerini sorgulamakta ve gerekli değişiklikleri yapmakta özgürdür.

12) Ekip, düzenli aralıklarla nasıl daha etkili olunacağını düşünür, ardından davranışını buna göre ayarlar ve ayarlar. Ekip kısa sürelerle toplanır, çalışma yöntemlerini gözden geçirir ve daha etkili çalışmak için geriye dönük (retrospective) toplantılar yaparak durumları gözden geçirir.

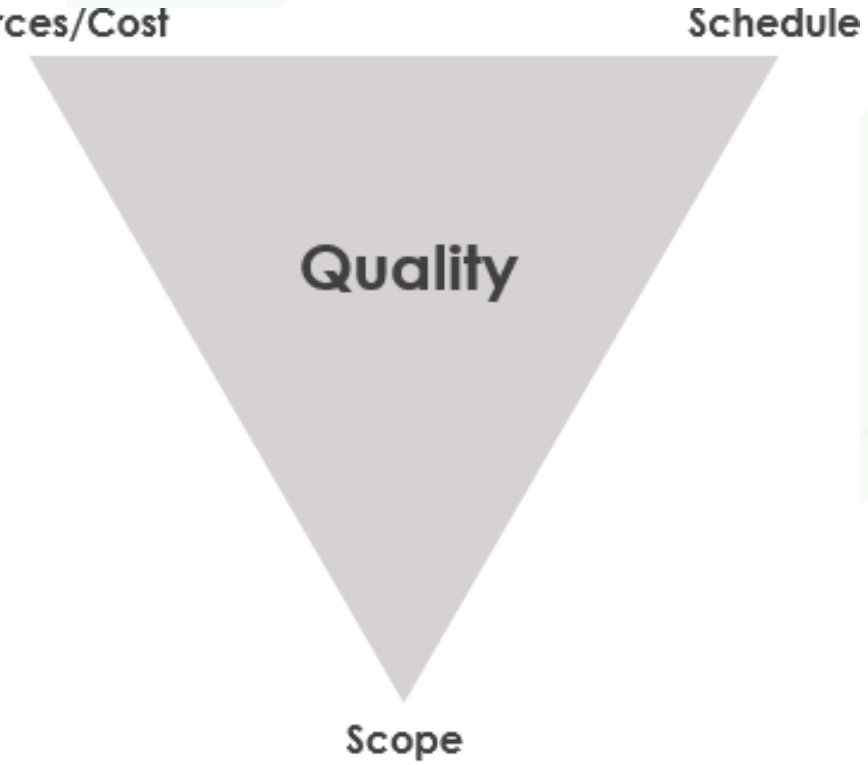
AGILE METHODOLGY VS WATERFALL



AGILE METHODOLGY VS WATERFALL



Waterfall: Scope is fixed, Cost and Schedules are variables

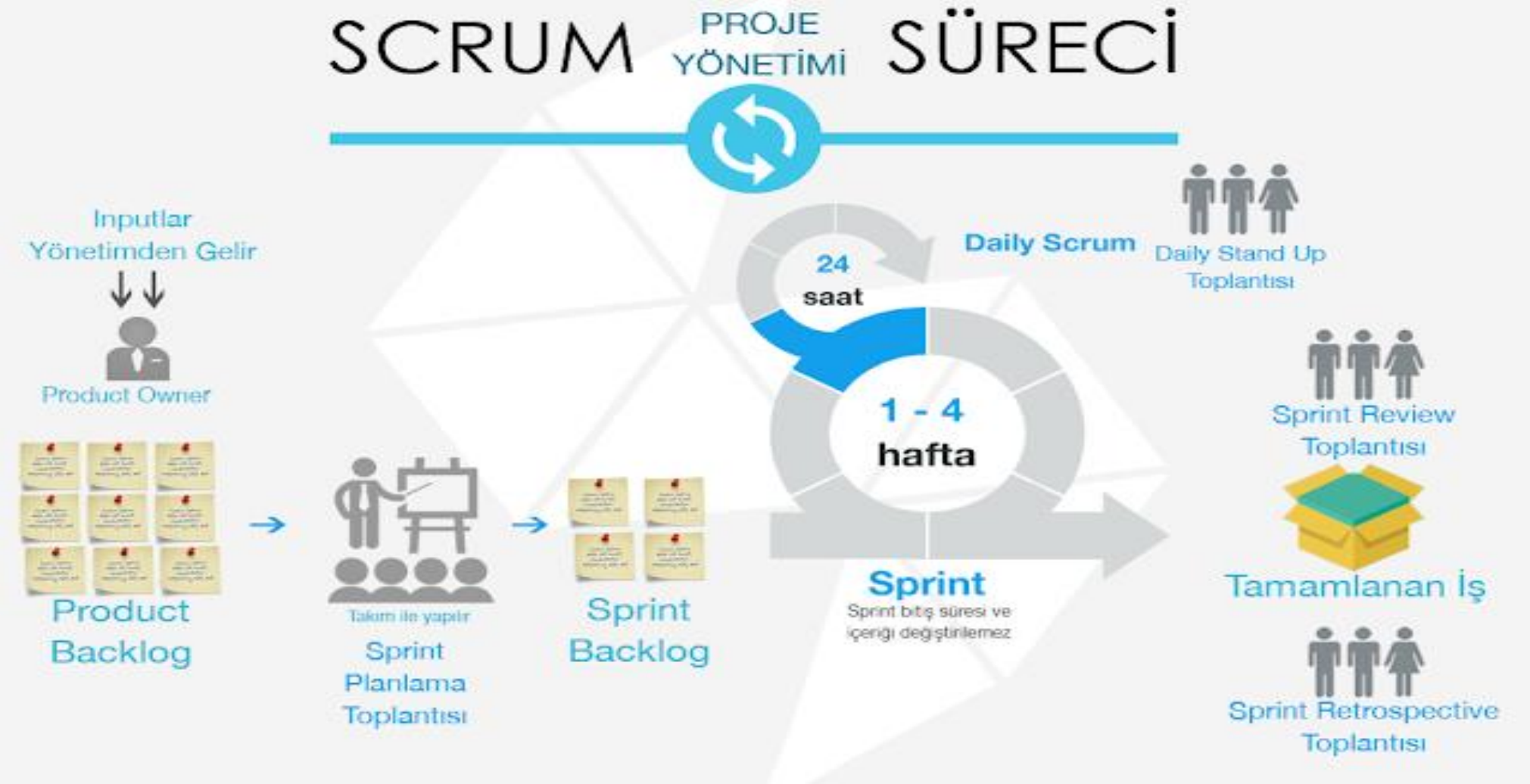


Agile: Cost and Schedules are fixed, Scope is variable

AGILE METHODOLGY VS WATERFALL

Method	Successful	Challenged	Failed
Agile	42%	50%	8%
Waterfall	26%	53%	21%

SCRUM

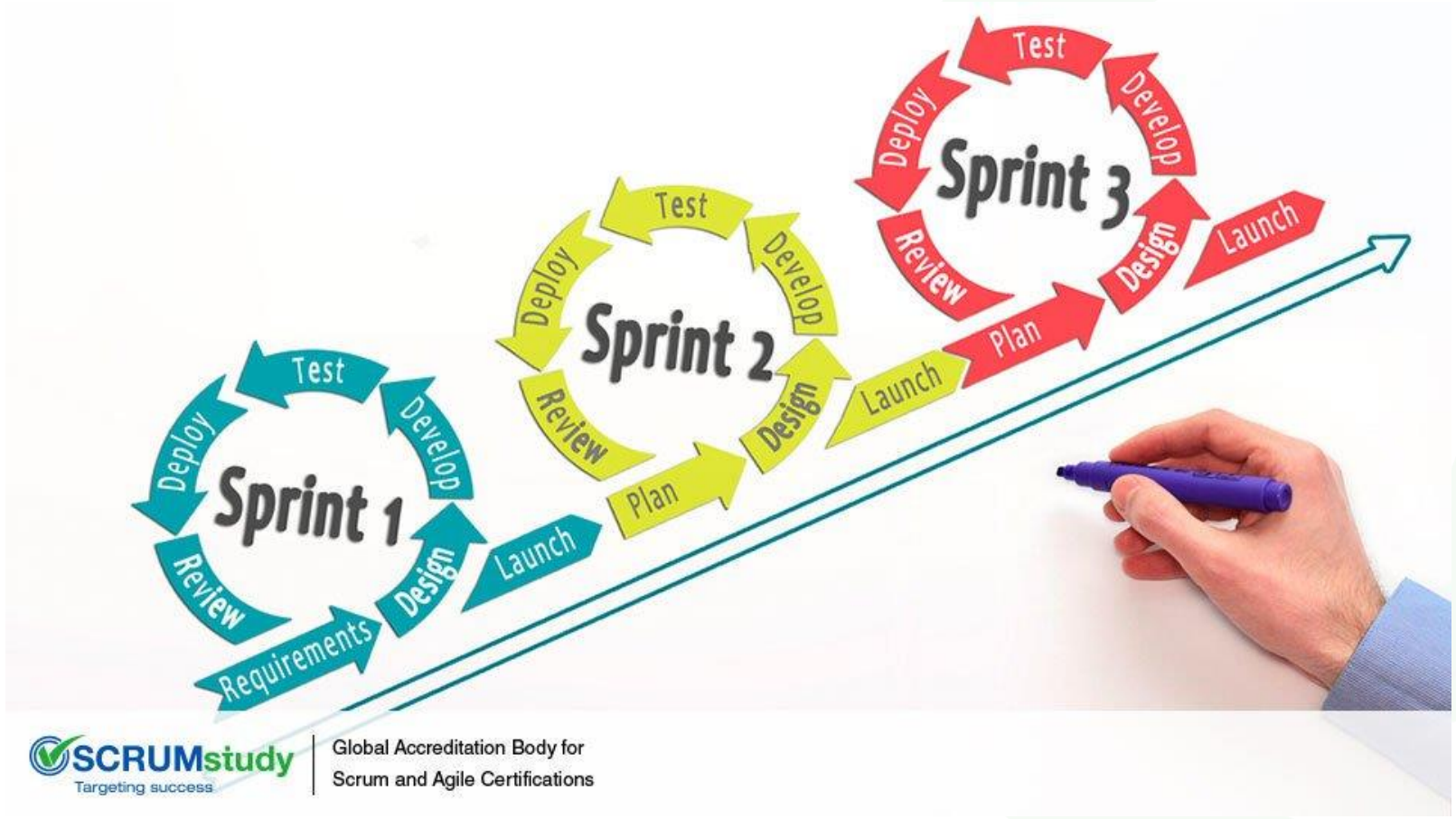

SCRUM PROJE YÖNETİMİ SÜRECİ
Inputlar
Yönetimden Gelir
Product Owner
Product
Backlog
Takım ile yapılır
Sprint
Planlama
Toplantısı
Sprint
Backlog
24
saat
1 - 4
hafta
Sprint
Sprint bitiş süresi ve
içeriği değiştirilemez
Daily Scrum
Daily Stand Up
Toplantısı
Sprint Review
Toplantısı
Tamamlanan İş
Sprint Retrospective
Toplantısı

SCRUM

Zaman içerisinde projelerin daha büyük ve karmaşık bir hal alması, bununla beraber müşterinin büyük resmi göremeyip gereksinimlerini tam olarak ortaya koyamaması, teknolojinin çok hızlı değişmesi ile beraber gereksinimlerin çabuk değişmesi ve bunu projemize entegre edemeyişimiz gibi problemlerden dolayı çoğu proje başarısızlık ile sonuçlanmaya başladı. Böylece proje sürecinin yönetilmesi konusu önemli bir konu oldu ve "Çevik (Agile) Yazılım Geliştirme Manifestosu" ortaya çıktı.

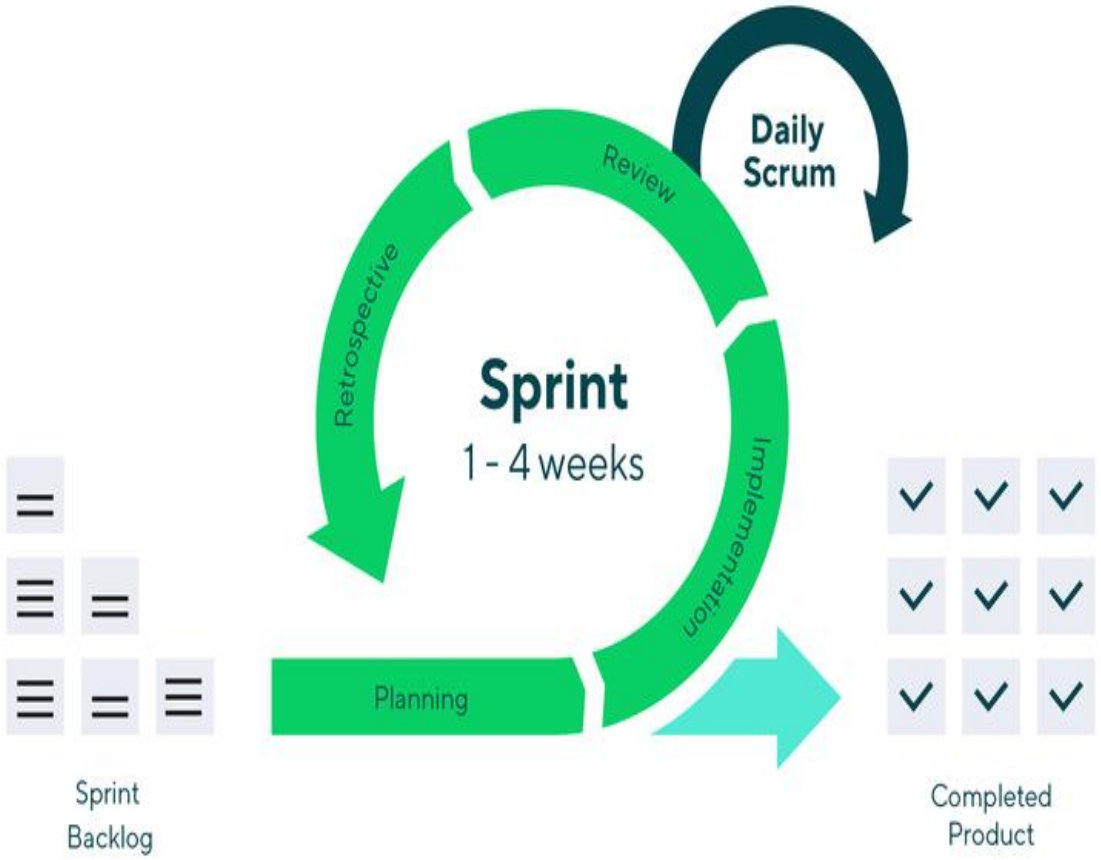
Scrum: Agile proje yönetim metodolojilerinden biridir. Kompleks yazılım süreçlerinin yönetilmesi için kullanılır. Bunu yaparken bütünü parçalayan; tekrara dayalı bir yöntem izler. Düzenli geri bildirim ve planlamalarla hedefe ulaşmayı sağlar. Bu anlamda ihtiyaca yönelik ve esnek bir yapısı vardır. Müşteri ihtiyacına göre şekillendiği için müşterinin geri bildirimine göre yapılanmayı sağlar. İletişim ve takım çalışması çok önemlidir.

SCRUM



"Rugby" yaklaşımı : "takımın, mesafenin tümünü hep beraber, bir birim halinde topu ileri geri atarak kat etmesidir."

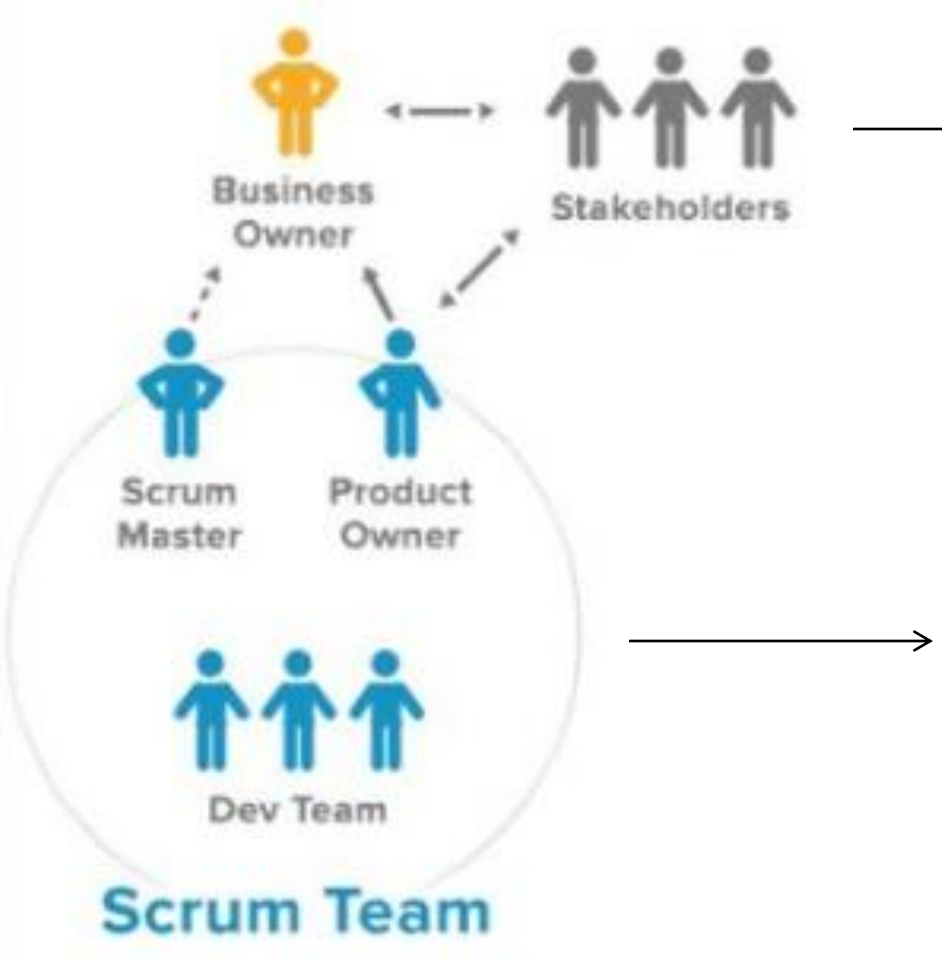
SCRUM



SPRINT

- Her sprint genellikle 2 ila 4 hafta veya en fazla bir takvim ayı sürer. Ürünleri her seferinde küçük bir parça oluşturmak, üretkenliği teşvik eder ve ekiplerin geri bildirime ve değişime yanıt vermesini ve tam olarak gerekli olanı oluşturmasını sağlar.
- Scrum'da ürün sprint'te tasarlanır, kodlanır ve test edilir.
- Yapilacak tum isler, Product Backlog da biriktirilir, Product Owner (PO) nun belirlediği önceliğe göre Sprint Backloguna alınır ve bir sprintte bitirilerek ürünün demosuna eklenir.

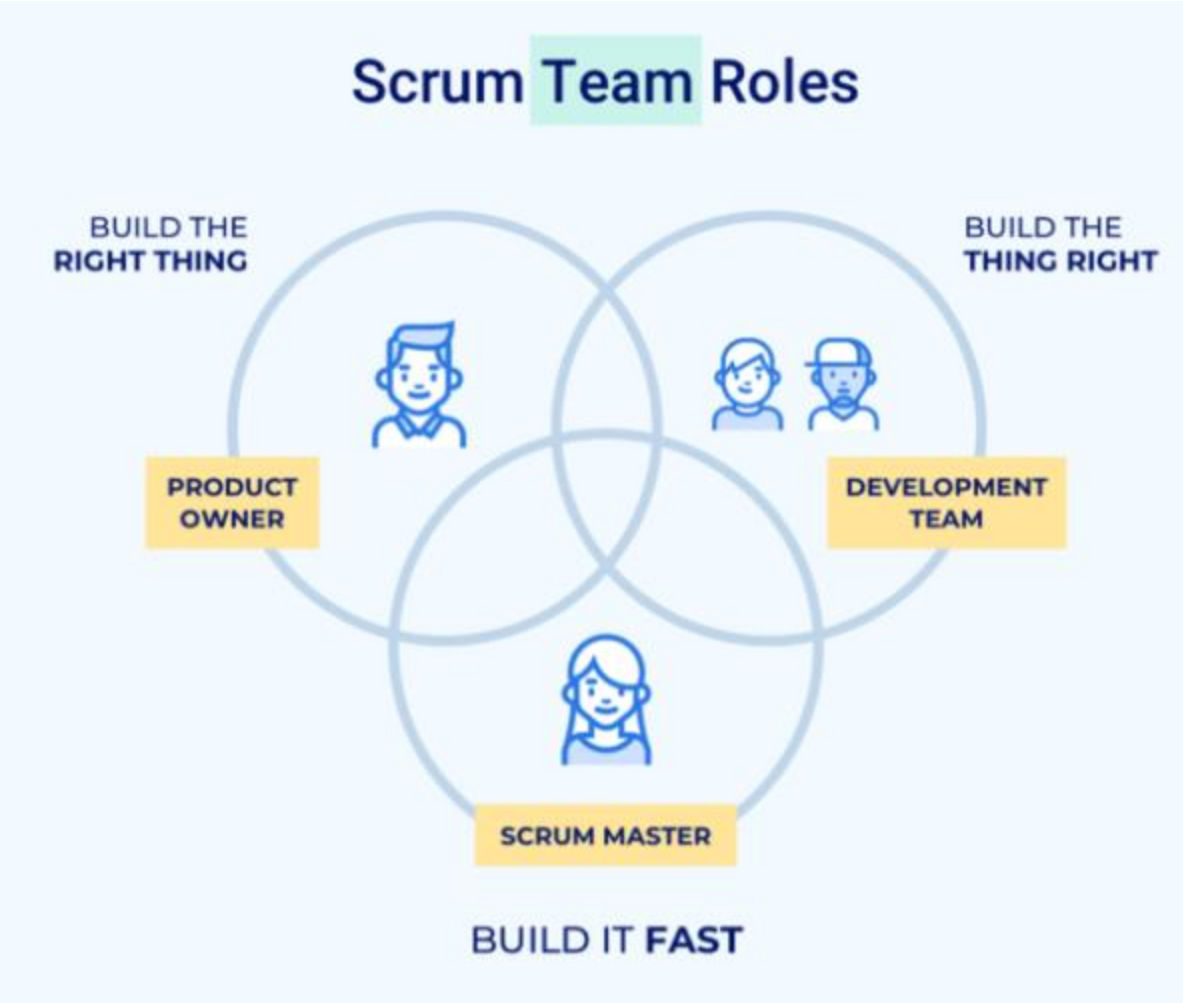
SCRUM TEAM



Chicken Roller: Scrum'ın işleyişinde aktif olarak yer almayan kişilerdir. Müşteriler, satıcılar gibi.

Pig Roller: Scrum sürecine dahil olanlar yani projede asıl işi yapan kişilerdir. Bunlar;
1) Product Owner (PO)
2) Scrum Master
3) Geliştirme Takımı

SCRUM TEAM ROLES



Takım kendi kendini örgütler. (Self Organized)
Böylece kendi içerisinde uyum içinde olan takımlar daha başarılı sonuçlar alırlar.

SCRUM TEAM ROLES

Product Owner:

Geliştirme takımı ve müşteri arasındaki iletişimi sağlar. Projenin özelliklerini tanımlar. Projenin önceliklerine göre Product Backlog (iş listesi) oluşturur. Ekipte Stakeholders'ı temsil eder.

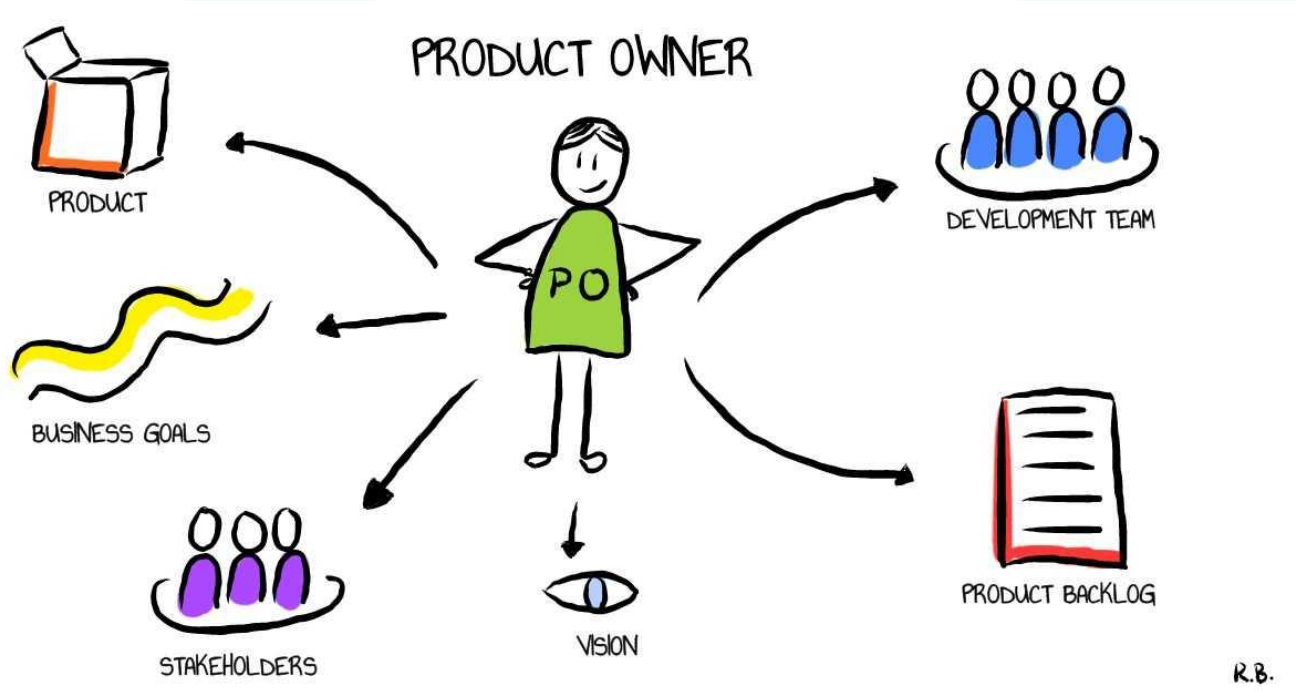
- İş Listesini (Product Backlog) yönetir. Birinci önceliği; iş listesi'ni yönetmektir. Product Backlog'i yönetmek demek, ürünü yönetmek demektir.
- Ürünle ilgili yapılacak bütün geliştirme Product Backlog'da bulunur.
- Product Backlog'un sahibi PO'dur.
- Product Backlog herkes tarafından erişilebilir ve anlaşılabilir olmalıdır.
- Product Owner, Geliştirme Takımı ve Stakeholders arasındaki iletişimi gerçekleştirir.
- Her Sprint'te hangi user story'lerin sprinte dahil edilecegine karar verir
- Sprint değerlendirme toplantılarının organizasyonunu yapar.
- Sprint değerlendirme toplantıları'nın sahibidir.

SCRUM TEAM ROLES

- Scrum takımın üyesidir
- İletişimi kuvvetlidir
- Müşteri ve Development Team

arasında iletişim kurar

- Sorumluluğu elinde tutar
- Belirleyicidir
- Kararlı ve ulaşılabilir
- Urun sorumluluğunu kabul eder
- Karar verme yetkisine sahiptir
- İş ve etki alanı yeterliliğine sahiptir.
- Sprint değerlendirme toplantılarının sahibidir.



SCRUM TEAM ROLES

Product Owner Kim Değildir :

- Development Team'in ve Scrum'in yöneticisi değildir.
- Geliştirme Takımı teknik konularda kendi kendini yönetir.
- PO, teknik bir geçmişe sahip olabilir. Bu, hangi işi kimin yapacağına karışabileceği anlamına gelmez.
- PO, iş, görev ataması yapamaz.
- İşin nasıl yapılacağına karışamaz!

SCRUM TEAM ROLES

2) Scrum Master:

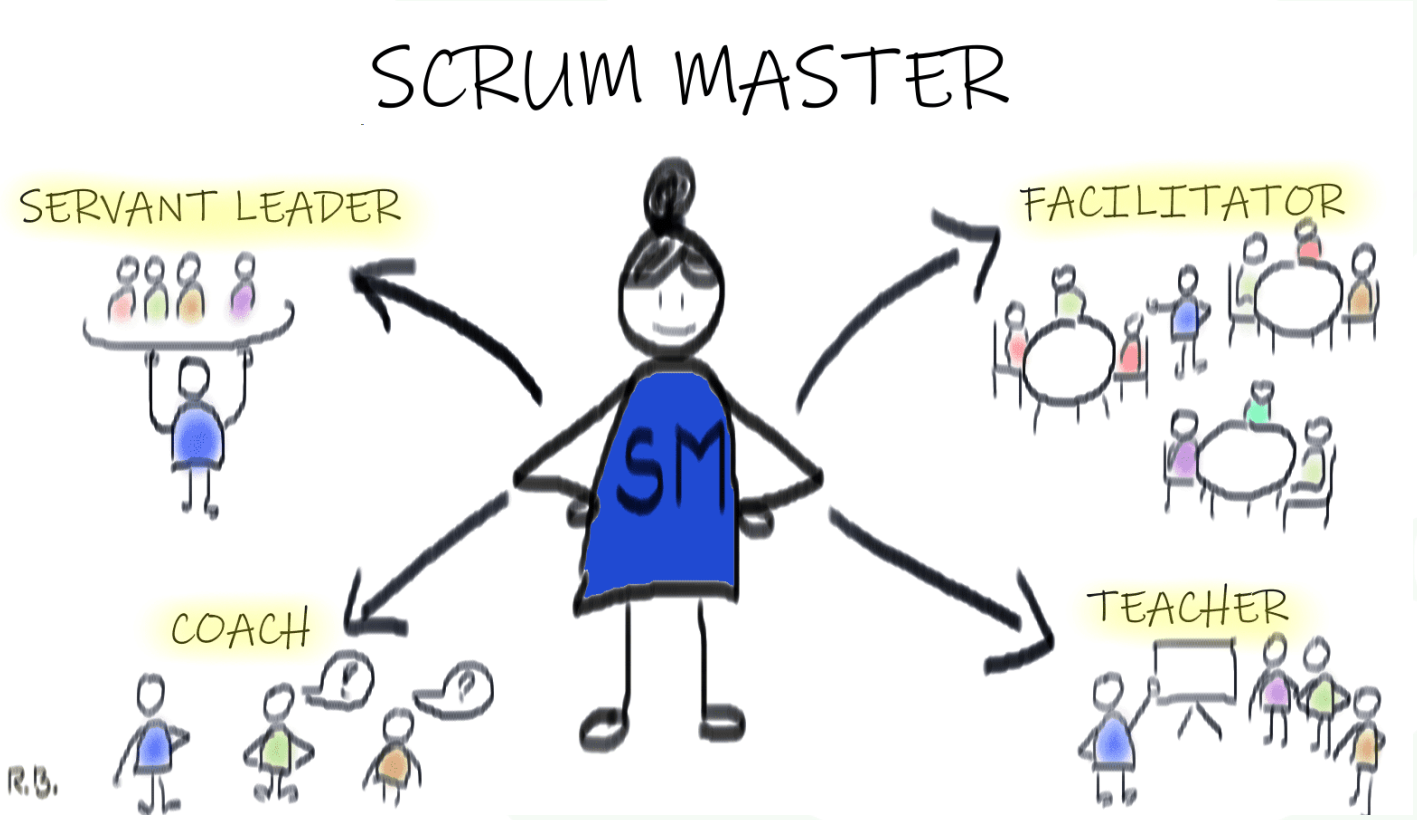
Scrum kurallarını, teorilerini ve pratiklerini iyi bilir ve takımın bu kurallarını uygulamasından sorumlu kişidir. Takımın yöneticisi değildir. Takımı rahatsız eden, verimli çalışmalarını engelleyen durumları ortadan kaldırır.

“Scrum Master, takımın Scrum değerlerine, pratiklerine ve kurallarına bağlı kalmasını garanti altına almakla sorumludur. Scrum Master, takımı ve organizasyonu Scrum’a adapte eder.”

SCRUM TEAM ROLES

2) Scrum Master:

- Scrum takımının üyesidir
- İş birlikçidir
- Koruyucudur
- Yardımcıdır
- Problem çözücüdür
- Kararlı ve ulaşılabilir
- Bilgilidir



SCRUM TEAM ROLES

2) Scrum Master:

- ScrumMaster, takıma rehberlik ve koçluk eder, karşılaşılan engelleri ortadan kaldırmalarına yardımcı olur. Takım içi harmoniyi, ekip elemanları arasındaki uyumu, iletişimi arttırmak için çabalar.
- Sprint Planlama, Sprint Retrospektif, Günlük Scrum ve Sprint Review gibi Scrum ritüellerini ve toplantılarını kolaylaştırır.
- ScrumMaster takımın güvenli ve sorunsuz bir ortamda çalışabildiğinden emin olmalı, gerektiğinde takım elemanlarına bireysel koçluk da dahil olmak üzere bir çok hizmetini sunmalıdır.

SCRUM TEAM ROLES

2) Scrum Master:

- Product Owner ile iliskileri yönetir. Scrum Master, ürün sahibi tarafından belirlenmiş işlerin takımdaki herkes tarafından anlaşıldığından emin olur; ürün sahibinin iş listesini etkili bir şekilde organize edebilmesi için teknikler bularak ona yardımcı olur.
- Değişime Liderlik Eder. Tüm bu görevlerin yanı sıra Scrum Master'lar değişime liderlik ederler.
- Scrum Master, Scrum'ın ve organizasyondaki çevik değişimin vücut bulmuş halidir.

SCRUM TEAM ROLES

3)Developmet Team:

Geliştirme Ekibi, Front-End Developer, Back-End Developer, Dev-Ops, QA (Tester), İş Analisti (BA), UI-UX designer vb. gibi özel becerilere sahip kişilerden oluşabilir.

- Product Owner'ınyapılacak işler listesi olan Product Backlog'tan, belli bir sürede (Sprint) yapabileceği kadar işi, Product Owner'ınbelirlediği önceliğe gore yapan geliştirme takımıdır.
- Teknik Konularda Sorumluluk Development Team'e aittir.
- PO ve SM development team'in yapacagi islerin önceliğini belirleyebilir ama neyi nasil yapacaklarina karışmazlar.

SCRUM TEAM ROLES

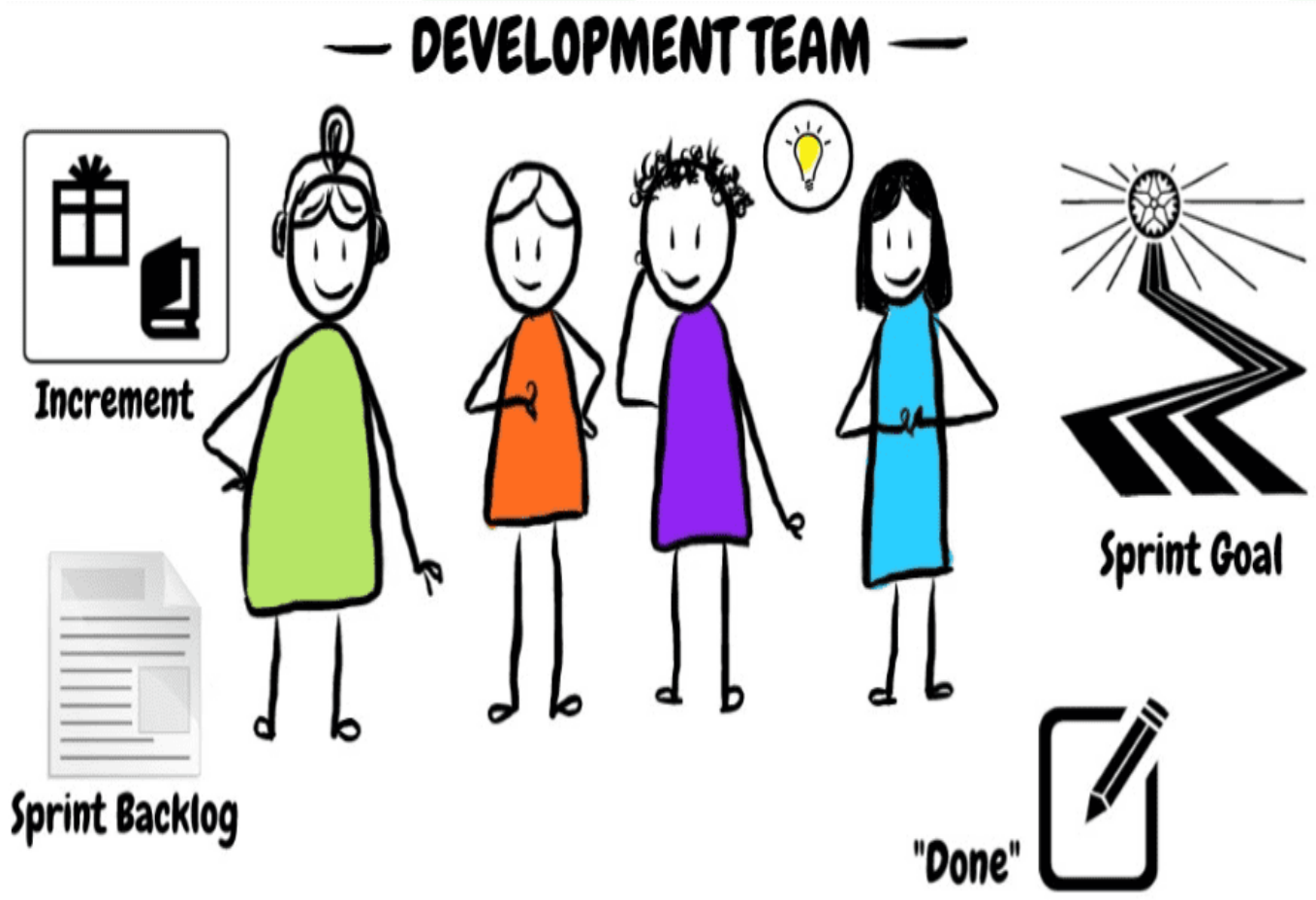
3)Developmet Team:

- Takım içerisindeki kişilerin rolleri ve yetenekleri ne olursa olsun, dışarıya karşı bir işin tamamlanmasından tüm takım sorumludur.
- Bir Sprint'e alınan bütün işleri tamamlayacak özelliklere sahip kişilerdir. sprint backlogu oluştururlar. Kendi kendini yönetir. İşin verilmesini beklemezler, işi kendileri alır ve geliştirirler.
- Development Team, Product Backlog'tançektiği bir Product Backlog Item'ı, Product Owner'ın önüne çalışan bir kod parçacığı olarak koymakla yükümlüdür.
- "Self Organize" olmalıdır. Development Team, sorumluluğunu aldığı işlerin yapılması için bir iş tanımlamasına veya bir iş takipçisine ihtiyaç duymaz.

SCRUM TEAM ROLES

3)Developmet Team:

- Scrum takımının üyesidir
- Çapraz Fonksiyonel: Dışarıdan herhangi bir yardıma ihtiyaç duymadan çalışmalarını tamamlamak için gerekli tüm beceri setlerine sahiptir.
- Kendi kendine yeterli
- Kendi kendine organize
- Yetenekli
- Kararlı ve ulaşılabilir
- Sorumlu



SCRUM PROJE YÖNETİMİ SÜRECİ

Inputlar
Yönetimden Gelir

Product Owner

Product Backlog

Takım ile yapılır

Sprint Planlama Toplantısı

Sprint Backlog

24 saat

Daily Scrum

Daily Stand Up Toplantısı

1 - 4 hafta

Sprint

Sprint bitiş süresi ve içeriği değiştirilemez

Sprint Review Toplantısı

Tamamlanan İş

Sprint Retrospective Toplantısı