SDLC

Software Development Life Cycle

(Yazılım Geliştirme Yaşam Döngüsü)

1. Ders
26/02/2022

Bu Dersten Sonra
Resume veya CV'nize Neler Yazabilirsiniz?

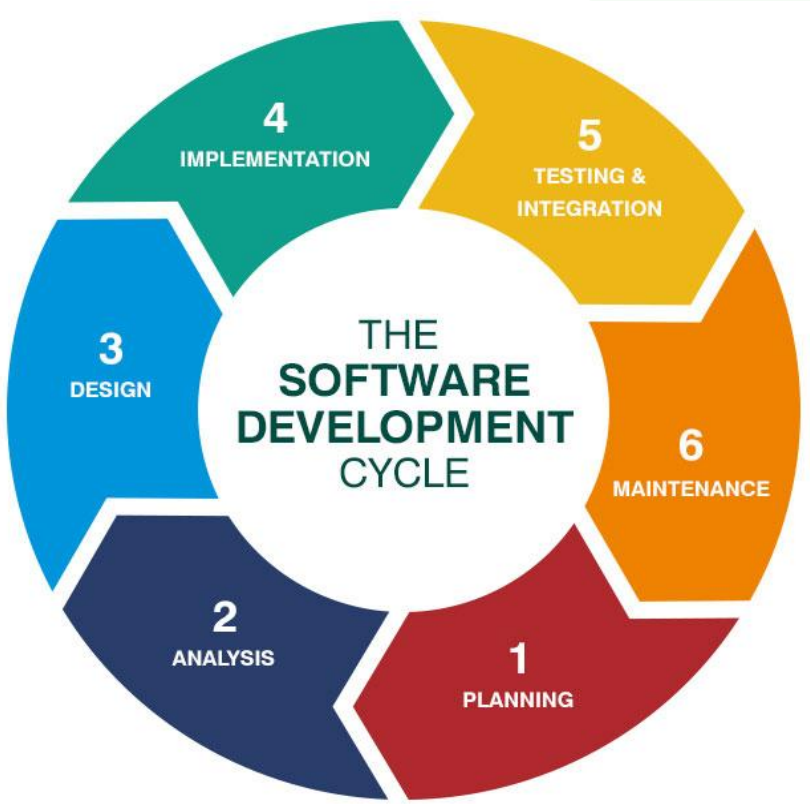
SDLC (Software Development Life Cycle),
Agile Methodolgy, WaterFall Methodology,
STLC (Software Testing Life Cycle),
BLC (Bug Life Cycle),
Project Managment Tools (Proje Yönetim Araçları),
Jira, Jira&Xray
konularında bilgili.

SDLC

Software Development Life Cycle
(Yazılım Geliştirme Yaşam Döngüsü)

SDLC, yazılım ürünlerini geliştirmek veya değiştirmek için planlanan yazılım geliştirme sürecidir. Yazılım ürünlerinin nasıl geliştirilmesi gerektiğini veya nasıl iyileştirilmesi gerektiğini anlatan ayrıntılı bir plan içerir.

SDLC, yüksek kaliteli (high quality) ve kullanıcı beklentilerini (user expectaion) karşılayan yazılımları tasarlamak, geliştirmek ve test etmek için yazılım endüstrisi tarafından kullanılan bir süreçtir.



NEDEN SDLC ?

DİJİTAL

Güvenlik açığı ile dünya çapında ses getiren Twitter'ın piyasa değeri düştü

Twitter'ın hisseleri borsa kapanışının ardından yüzde 4 düşüş gösterdi.

Tuğçe İçözü
16 Temmuz 2020

Twitter Support
@TwitterSupport

We are aware of a security incident impacting accounts on Twitter. We are investigating and taking steps to fix it. We will update everyone shortly.

ÖÖ 12:45 · 16 Tem 2020 · Twitter Web App

30 B Retweet **7.288** Alıntı Tweetler **111,7 B** Beğeni

Boeing'in önce uçakları sonra hisseleri düştü 25 milyar dolar kaybetti

Paza günü Etiyopya'da 157 kişinin ölümüne neden olan Boeing 737 MAX uçak kazası sonrası ülkeler bir bir uçuşlarını durdururken şirketin hisseleri borsada yere çakıldı.

NEDEN SDLC ?

[Fotoğraf: AA]

ABONE OL Google News Takip et: @trthaber

Sosyal medya platformları Whatsapp, Instagram ve Facebook'a dünya genelinde yaşanan erişim sorunu devam ediyor. Facebook'un hisseleri yüzde 5'in üzerinde düştü.

Yemek Sepeti çöktü: Hem internet sitesine, hem de uygulamaya ulaşılamıyor

18:37 10.01.2021 (güncellendi: 04:48 11.01.2021)

© Fotoğraf

Abone ol Google News

Hem yemeksepeti.com hem Yemek Sepeti IOS ve Android uygulamalarına erişim sağlanamadı. Firmadan henüz bir açıklama yapılmadı.

NEDEN SDLC ?

Ünlü Alışveriş Sitesi Amazon Çöktü, Ürünler 3,6 Kuruştan Satıldı

Haberler.Com - Haberler | Güncel
16 Aralık 2014 Salı 13:02

ABD'li online alışveriş devi Amazon.com'da yaşanan bir hacker skandalı nedeniyle 20 bine yakın ürün, 3,6 kuruştan satıldı.

Ünlü alışverisş sitesi Amazon.com'un **İngiltere** operasyonunda yaşanan bir hacker skandalı, 20 bine yakın ürünün 1 penny'lik (yaklaşık 3.6 kuruş) fiyattan satılmasına yol açtı.

Akbank'tan yeni açıklama! Akbank sistem arızası düzeldi mi, sorun devam ediyor mu?

Akbank kullanıcıları salı gününden bu yana mobil ve internet bankacılığında sorun yaşıyordu. Öte yandan ATM ve pos cihazlarında da hata olduğu öne sürüldü. 'Oturum kapandı' sorunu ile karşılaşan vatandaşlar Akbank internet bankacılığı çöktü mü, siber saldırıya mı uğradı sorularını gündeme getirdi. Akbank'tan konu hakkında açıklama geldi.

• 09 Temmuz 2021 - 00:42 • Son Güncelleme: 09 Temmuz 2021 - 00:42

Software Hatalarının Sonuçları 1

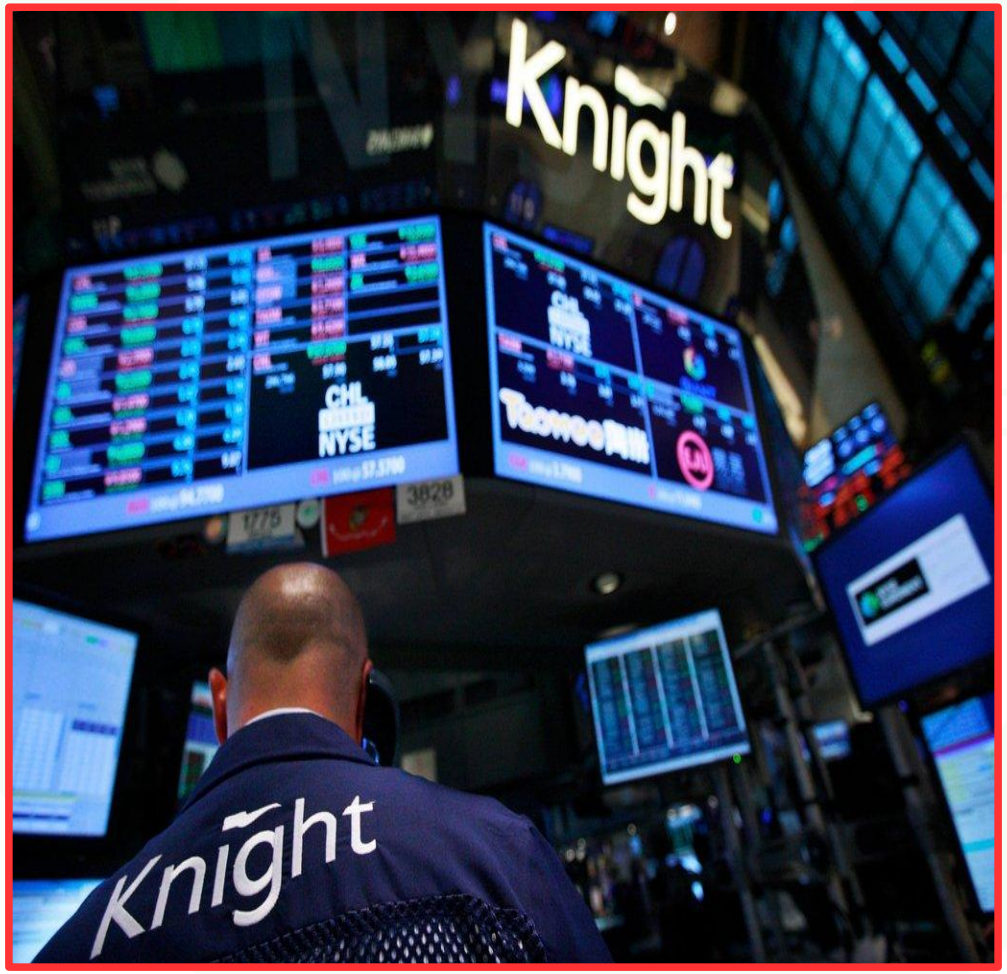
- 4 Haziran 1996'da fırlatılan Ariane 5 uzay aracını kodlarken, Ariane 4' te kullanılan bir modül düzgün test edilmeden yeniden kullanılmıştı ve Ariane 5'e uyum sağlamamıştı.
- Parçalanmanın sebebi bir yazılım hatasıydı. 64 bitlik ondalıklı sayı, 16 bit işaretli tam sayıya çevrilirken bulunan sonuç beklenenden büyük çıkıyordu.
- O gün fırlatma için geri sayım yapıldı ve roketin motorları ateşlenerek kalkış başladı. Hızlanarak yoluna 37 saniye boyunca devam eden Ariane 5 roketi; o saniyeden sonra yanlış yöne doğru 90 derece dönmeye başladı. Bu durum, roketin kendini imha etme mekanizmasını tetikledi.
- Bu kaza, **370 milyon dolara** mal oldu.
- CNES ekibi bu kazayı bir ders olarak almış, Ariane 502 uçuşunda, yazılımla ilgili yapılabilecek dönemin en kapsamlı çalışmasını yürütmüşlerdi.

Software Hatalarının Sonuçları 2

- Therac-25 cihazı, kanser hastalarının tedavisinde kullanılmak için tasarlanmıştı.
- Yapılan değişiklik Therac-20'de güvenlik önlemi olarak bulunan elektromekanik güvenlik kilitlerinin yazılımsal güvenlik önlemleriyle değiştirilmesiydi.
- Ne yazık ki gelişim olarak görülen bu hata, Therac-20'nin kodlarında bulunan ancak fark edilemeyen hatanın, Therac-25'te ortaya çıkmasına sebep oldu.
- Bu hata (bug) yüzünden yaklaşık 5 hastanın ağır dozda radyasyon sebebiyle hayatını kaybettiği raporlandı.

Software Hatalarının Sonuçları 3

- New York borsasında piyasaları yönlendiren en büyük şirketlerden biri olan Knight Capital, 1 Ağustos 2012'de yeni bir yazılım güncellemesi yapma kararı aldı.
- Saat 09.00 sularında New York Borsası işlemler için açıldı ve Knight Capital'ın yatırımcıları, varlıklarını satmak veya almak için talimat verdi.
- Yalnızca 45 dakika sonra; Knight Capital'ın sunucuları, 4 milyon işlem gerçekleştirdi.
- Olay, bir teknisyenin yeni Perakende Likidite Programı (RLP) kodunu, Knight Capital'in hisse senedi siparişleri için otomatik yönlendirme sistemi olan sekiz SMARS bilgisayar sunucusundan birine kopyalamayı unutması sonrasında meydana geldi.
- Bu hata, şirkete **460 milyon dolar** kaybettirerek iflasın eşiğine getirdi.



Software Hatalarının Sonuçları 4

Mars Climate Orbiter Hatası (23 Eylül 1999):

- Gezegenler arası ilk iklim uydusu olarak 1997'de fırlatıldı.
- Mars Orbiter, 1999'da Mars'ın yörüngesinde kayboldu.
- Kazanın yazılımda kullanılan İngiliz ölçü birimlerinin metrik sisteme yanlış çevrilmesinden kaynaklandığı belirtildi.
- NASA'da bir ekip hesaplarında İngiliz ölçü birimini (inç) kullanırken, projeye katılan diğer ekipse metrik (cm) sistemi kullanmıştı.
- **125 milyon dolarlık** uydu yörüngeye sabitlenmeye çalışırken Mars'a olması gerekenden fazla yaklaşarak imha olduğu düşünülüyor.

Software Hatalarının Sonuçları 5

- 1991 yılının şubat aylarında gerçekleşen Körfez Savaşı sırasında ABD'nin Suudi Arabistan'ın Zahran şehrindeki üssünde bir patlama yaşandı. Patlamanın sebebi ise üste bulunan anti balistik füze sisteminin doğru çalışmamasıydı.
- Yapılan sorgulamaların ve araştırmaların sonucunda patlama sebebinin üste bulunan anti balistik füze sisteminin bir yazılım hatası yüzünden ateşlenmemesi olduğu anlaşıldı.
- Bir insan için inanılmaz küçük olan 0,33 saniye, Al Hussein füzesini takip etmek için yapılan bir sistem için inanılmaz büyük bir hataydı.
- MIM-104 Patriot, havada bir cisim olduğunu algılamayı başardı ancak hata yüzünden cismi takip edemedi ve bunun bir füze olduğunu anlayamadı. Engellenemeyen füze yüzünden üste bulunan 28 asker hayatını kaybetti.

Software Hatalarının Sonuçları 6

- Toyota; 2014 yılında 160.000 Prius Hybrid aracının motorunu durduran yazılım hatasından dolayı, araçları geri çağırmak zorunda kalmıştı.
- 2007 yılında L.A. Havalimanı'nda meydana gelen network hattındaki arıza yüzünden, 8 saat süreyle iniş ve kalkışlar durdurulmuştu.
- 2018 yılında, Toyota, ABD, Avrupa ve Japonya pazarlarındaki toplam 2.43 mliyon hibrit aracını geri çağırdığını duyurdu.
- **Kara Pazartesi 1987:** Hisselerin otomatik satış talimatlarını yanlış zamanda tetikleyen yazılım, gün içerisinde borsanın ilk açıldığı yer olan Hong Kong'dan başlayıp bir günde Dow Jones'u %23, S&P 500'ü %20 düşürerek hala kırılmamış bir rekora imza atmıştır. (1987)
- **İntel İşlemci Bölme Sorunu (1993):** İntel İşlemcilerin ondalık sayıların bölüm sonuçlarında 0.006'lık bir sapma ile hata yapması sattığı 5 milyon chip için 475milyon$'ına mâl oldu.

NEDEN SDLC ?

- Internet browserların yıllara göre kullanımları
https://www.visualcapitalist.com/internet-browser-market-share/

- Dünyada en çok ziyaret edilen 50 site
https://www.visualcapitalist.com/the-50-most-visited-websites-in-the-world/

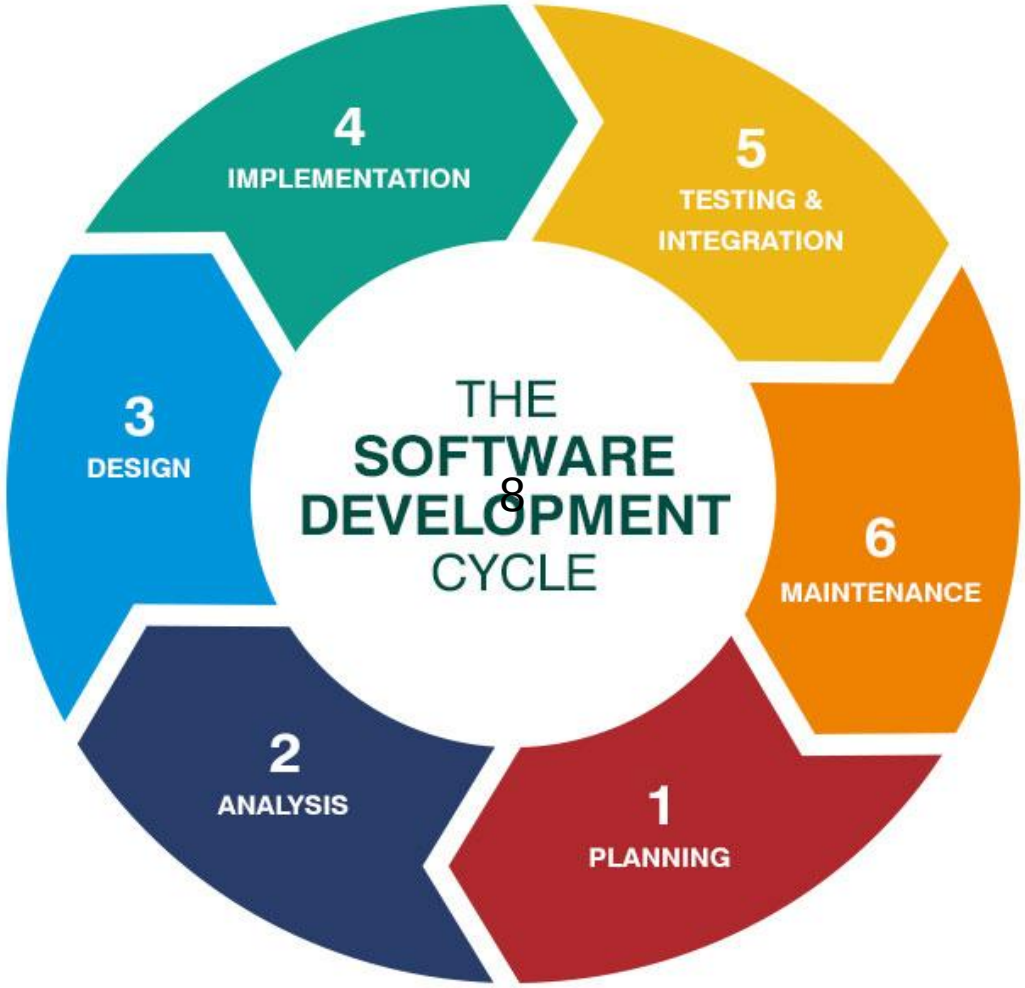
- Dünyanın en zengin kişileri
https://www.visualcapitalist.com/top-10-richest-people-worldwide-2021/

User Expectation: Kullanıcı Beklentisi
High Quality: Yüksek Kalite

SDLC'NİN FAYDALARI

- Projenin takibini ve kontrolünü sağlar.
- Tüm planlama ve process'in, yatırımcılar ve işveren tarafından görülebilmesine imkan tanır.
- Yapılan planlama ve toplantılar, projenin oluşturma ve geliştirme hızını artırır.
- Tüm ekibin iletişimini güçlendirir.
- Projenin risklerini azaltir.
- Doğru yapılan SDLC , en yüksek düzeyde yönetim kontrolüne ve dokümantasyona izin verir.
- Çalışanlar neyi, neden yapmaları gerektiğini anlarlar.
- Tüm taraflar hedef üzerinde önceden hemfikirdir ve bu hedefe ulaşmak için net bir plan görür.
- Tüm taraflar, gerekli maliyetleri ve kaynakları görebilirler.

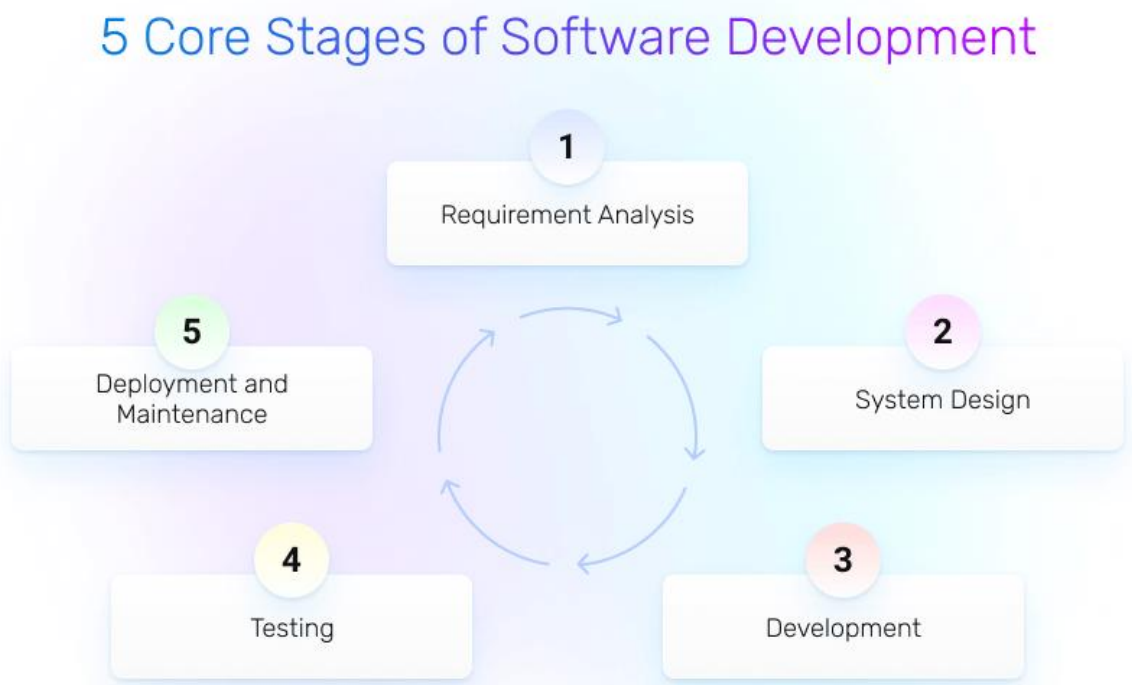
SDLC


4
IMPLEMENTATION
5
TESTING &
INTEGRATION
3
DESIGN
THE
SOFTWARE
DEVELOPMENT
CYCLE
6
MAINTENANCE
2
ANALYSIS
1
PLANNING

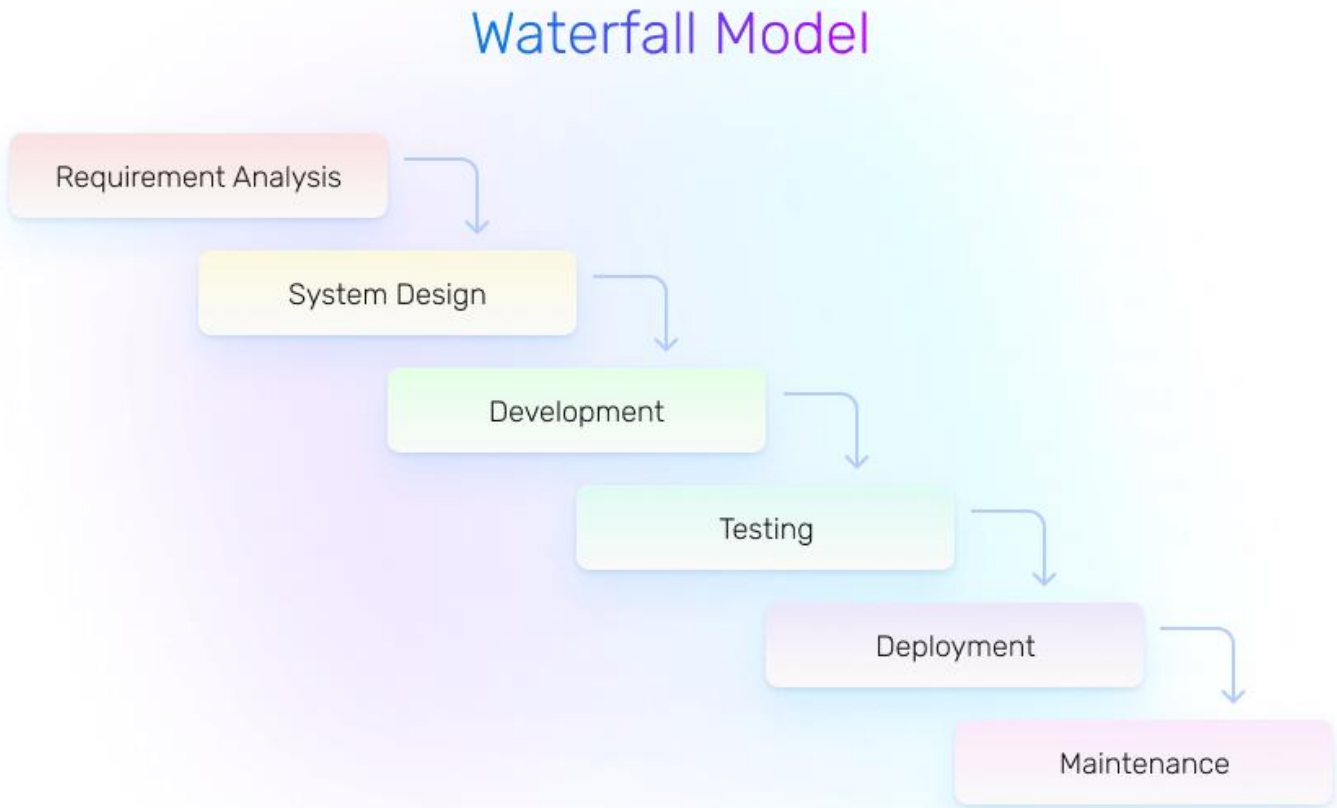

4
Kodlama
5
Test &
Entegrasyon
3
Tasarım
Yazılım
Geliştirme
Süreçleri
6
Bakım
2
Planlama ve
Analiz
1
Talepleri Doğru
Belirleme

SDLC Model ve Metodolojileri

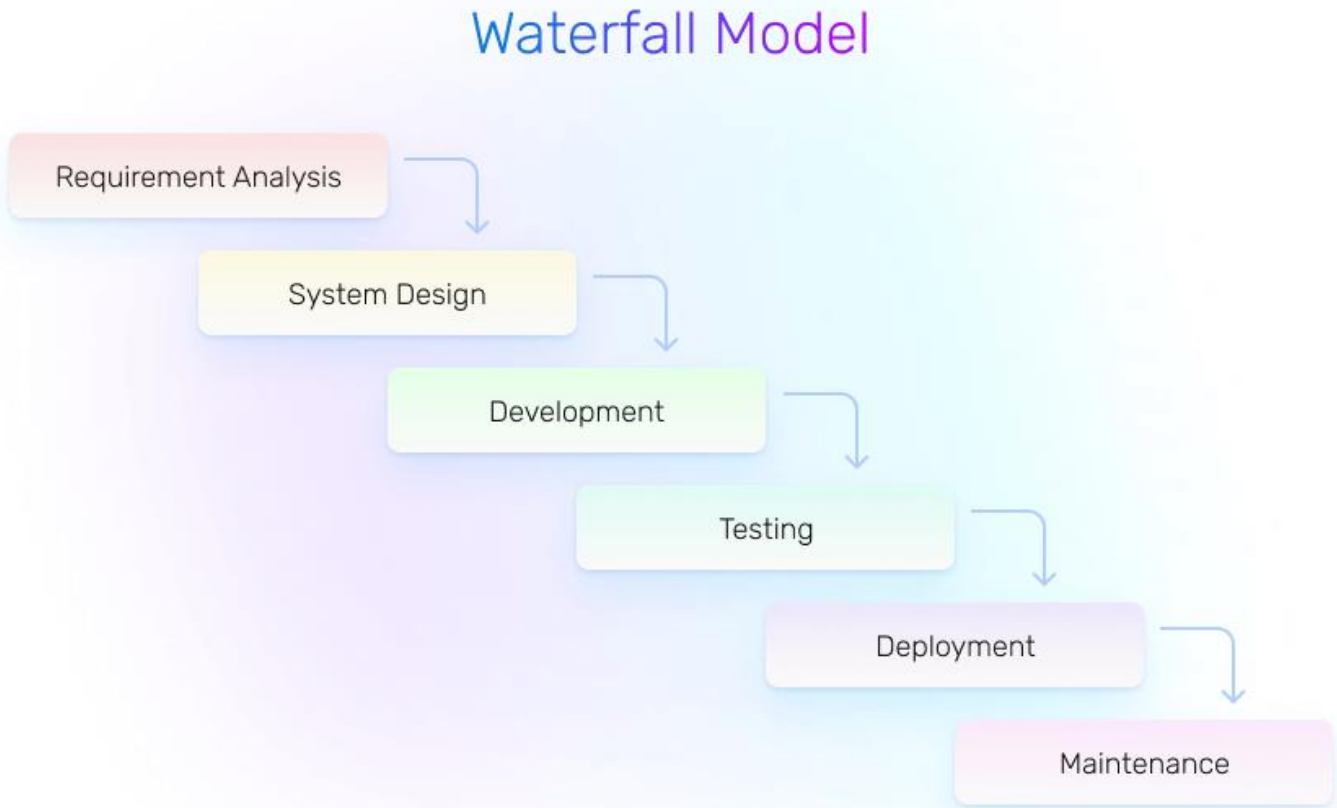
- Waterfall
- Agile
- Iterative
- Spiral
- V-shaped
- Scrum
- XP
- Kanban



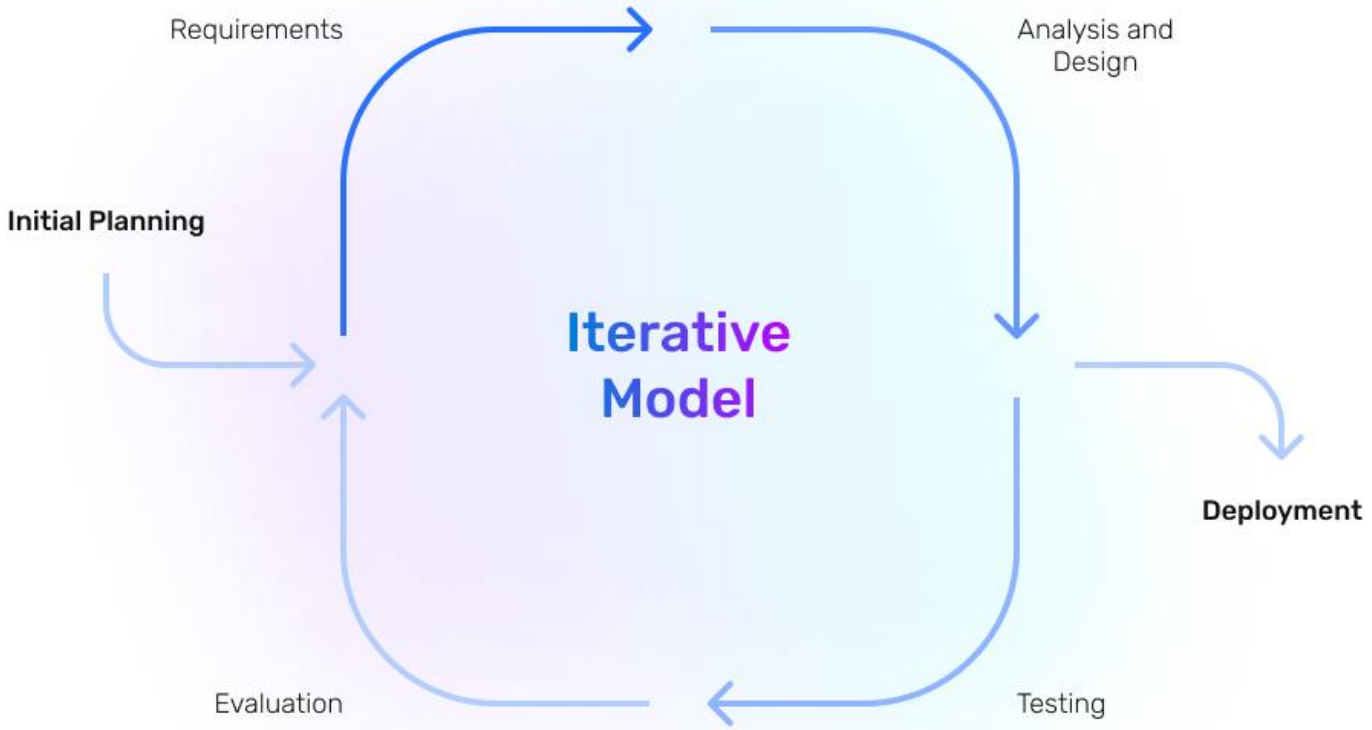
Waterfall Model



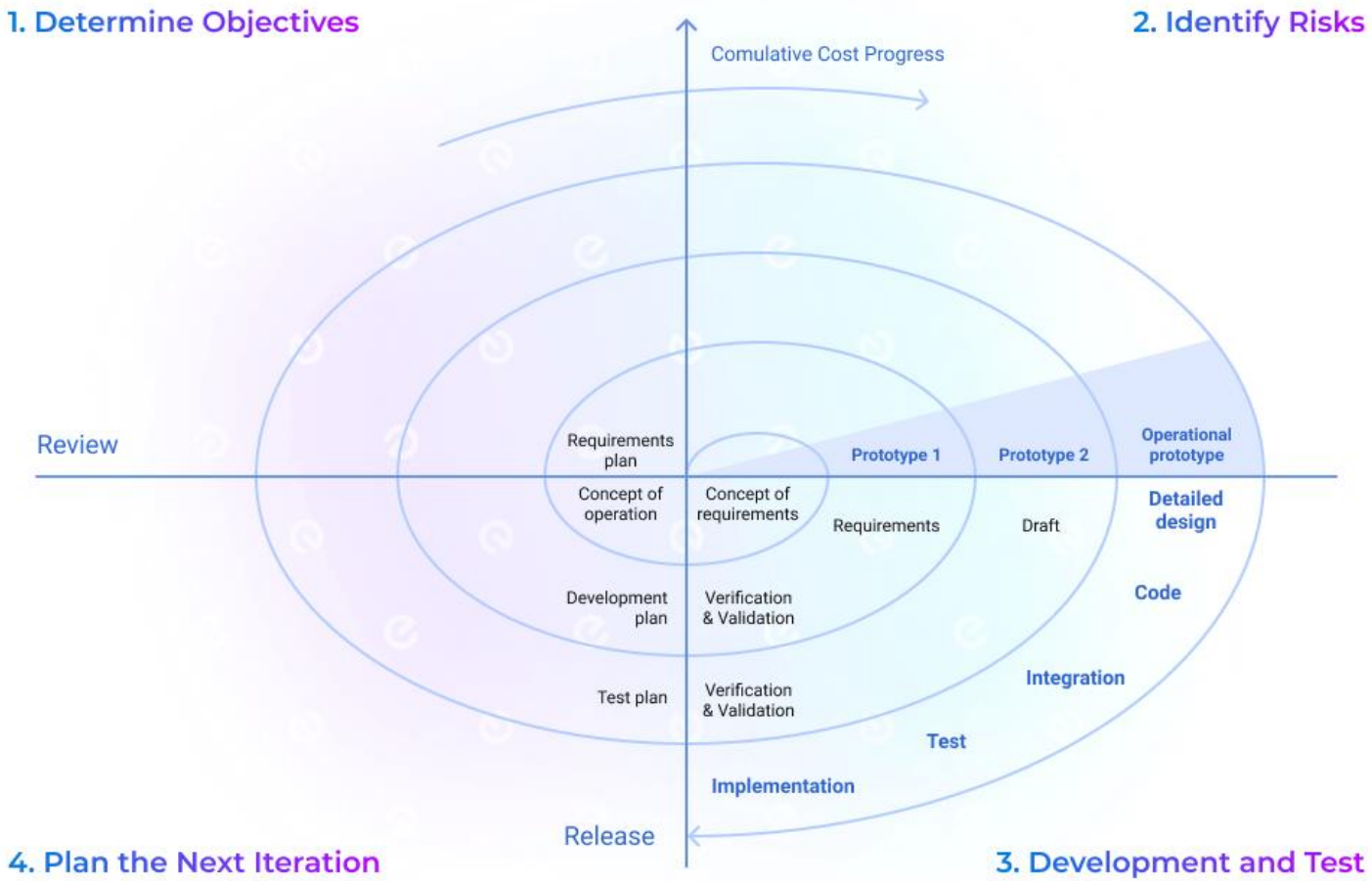
Waterfall Model



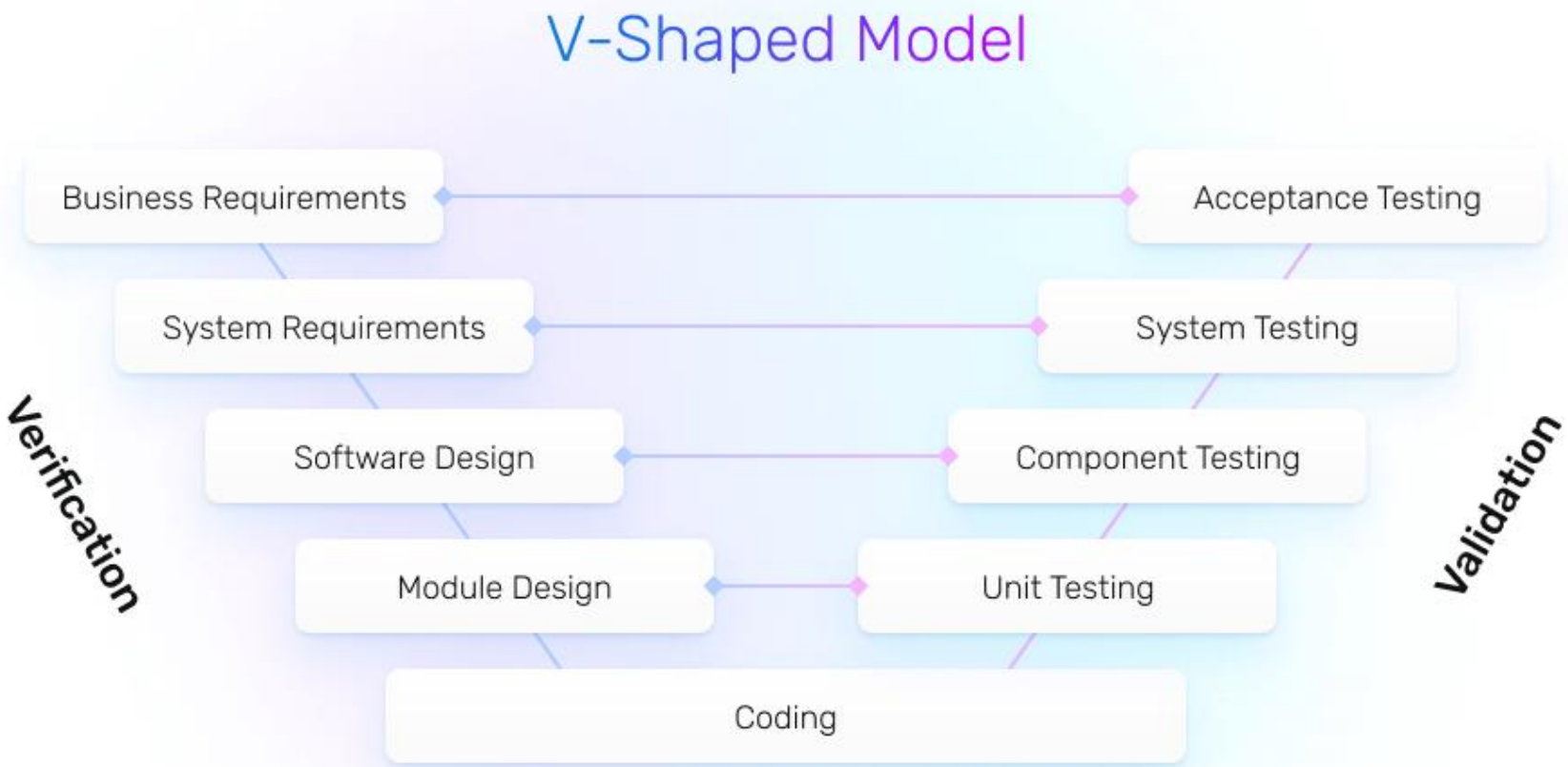
Iterative Model



Spiral Model



V-Shaped Model



SOFTWARE DEVELOPMENT PHASES
(Yazılım Geliştirme Aşamaları)

1) Planlama (Planning): SDLC'nin ilk aşaması planlamadır. Fizibilite çalışmaları yapılır. Projenin ihtiyaçları belirlenir, maliyetler hesaplanır, projenin faydaları ve riskleri hesaplanır. Müşteri ile birlikte gereksinimler belirlenir.

Başka bir deyişle, ekip projenin fizibilitesini ve projeyi en düşük riski göz önünde bulundurarak nasıl başarılı bir şekilde uygulayabileceğini belirlenir.

SOFTWARE DEVELOPMENT PHASES
(Yazılım Geliştirme Aşamaları)

2) Analiz (Analysis): Sistemin işlevlerinin ve gereksinimlerin ayrıntılı olarak incelenmesidir. Bu aşamada elde edilen veriler doküman haline getirilir. Ayrıca bu aşamada ekip çalışması çok önemlidir.

Müşteri (steakholder), developer, sistem analisti (Tester, QA), iş analisti (Business Analyst, BA), Proje yöneticisi (Project Manager, BM) vb. rollere sahip kişiler bir araya gelerek çalışmayı sürdürür.

Ekip arası iletişimin iyi olması, çalışma ortamında doğru kararlar verilmesine ve projenin iyi yönetilmesine yardımcı olur.

SOFTWARE DEVELOPMENT PHASES
(Yazılım Geliştirme Aşamaları)

3) Tasarım (Design): Bu aşamada gereksinimlerinin analiz edilmesiyle yazılım sistemi tasarlanır. Tasarım aşamasında kodlama yapmak söz konusu değildir.

Analiz aşamasında problemin ne olduğu belirlenirken tasarım aşamasında problemin nasıl çözüleceği belirlenir. Gereksinimleri karşılayan yazılım ürününün özellikleri , arayüzler ve yazılım ürününün faydaları , yetenekleri belirlenir.

Mimari tasarımda yazılım ürününün genel bir planı yapılır ve modüller belirlenir. Ayrıntılı tasarımda yazılımda kullanılacak algoritmalar, programlama dilleri , veritabanları ve bunun gibi detaylar belirlenir.

SOFTWARE DEVELOPMENT PHASES
(Yazılım Geliştirme Aşamaları)

4) Kodlama (Implementation): Kodlama sürecinin başladığı aşamadır. Bu aşamanın önemli noktalardan birisi doğru kodlama yapılmasıdır.

Doğru kodlama şekli bir başkasının da rahatça okuyabileceği ve bakım yapabileceği kodlar yazmaktır. Bu kodlama biçimi temiz kodlama (clean code) olarak adlandırılır.

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

5) Test (Testing): Bu aşamada gerçekleştirilmesi gereken önemli noktalardan biri, test etmektir.

Kodlama yapılırken ve kodlama sonrasında birçok test yapılır. Bu testlerden bazıları ; birim testleri, zorlanım-performans testi , yanlış değer testleri, tümleyim testi, kullanım senaryo testleri, yük testleri, kullanıcı kabul testi gibi testlerdir.

Ayrıca analiz aşamasından itibaren testler yapılması yazılım ürününde hata oranını azaltacağı için kaliteyi arttırır , maliyetleri (para, zaman vb.) azaltır. Bu yaklaşım erken test yaklaşımı (early testing) olarak adlandırılır.

SOFTWARE DEVELOPMENT PHASES
(Yazılım Geliştirme Aşamaları)

6) Teslim ve Bakım (Maintenance): Tüm aşamaların tamamlanmasının ardından yazılım ürünü müşteriye teslim edilir.

Ürün teslim edilirken ürünle ilgili bilgiler, kullanım kılavuzu da müşteriye teslim edilir.

Ürünün teslim edilmesi ile birlikte bakım aşaması başlar ve yazılım ürününün ömrü süresince devam eder.

Bu süreçte hata giderme, altyapıları iyileştirme , ürüne yeni özellikler ekleme gibi bakım faaliyetleri yapılır.

SDLC TEAM

1) Project Manager (PM)
   Proje Yöneticisi

2) Business Analyst (BA)
   İş Analisti

3) Developer (Dev)
   Yazılımcı

4) Quality Analyst (QA)
   Kalite Analisti (Tester)

PROJECT MANAGER (PM)

Project manager: Proje yöneticisi, takımdaki herkesin rolünü bilmesini ve yerine getirmesini ve bu rollerin gerçekleştirileceği inancına göre hareket etmesini sağlar.

- Proje planının geliştirilmesinden sorumludur
- Proje sahipleri (Stakeholder) ile yakın ilişki kurar
- Takım içerisindeki iletişimi sağlar
- Proje riskini yönetir
- Proje çizelgesini hazırlar
- Proje bütçesini yönetir
- Projede çıkabilecek karışıklıkları (conflicts) önler (Kriz yönetiminden sorumludur)
- Görev dağılımını yönetir.

BUSINESS ANALYST (BA)

Şirketlerin, iş süreçlerini değerlendirme, gereklilikleri öngörme, iyileştirme alanlarını açığa çıkarma ve çözümler üretme faaliyetlerini yürütür. Bir proje veya programın ihtiyaçlarını belirleyerek, bunları yönetici ve ortaklara iletir. İş sorunlarına teknik çözümler geliştirmek için çalışır.

- Business sorunları ve teknoloji çözümleri arasında bir köprü vazifesi görür
- Requirements (Gereksinimler) yönetimini ve iletişimini sağlar.
- Alınan kararların anlaşılır bir dile dökülmesini sağlar.
- **Business Requirement Document (BRD)** oluşturur.

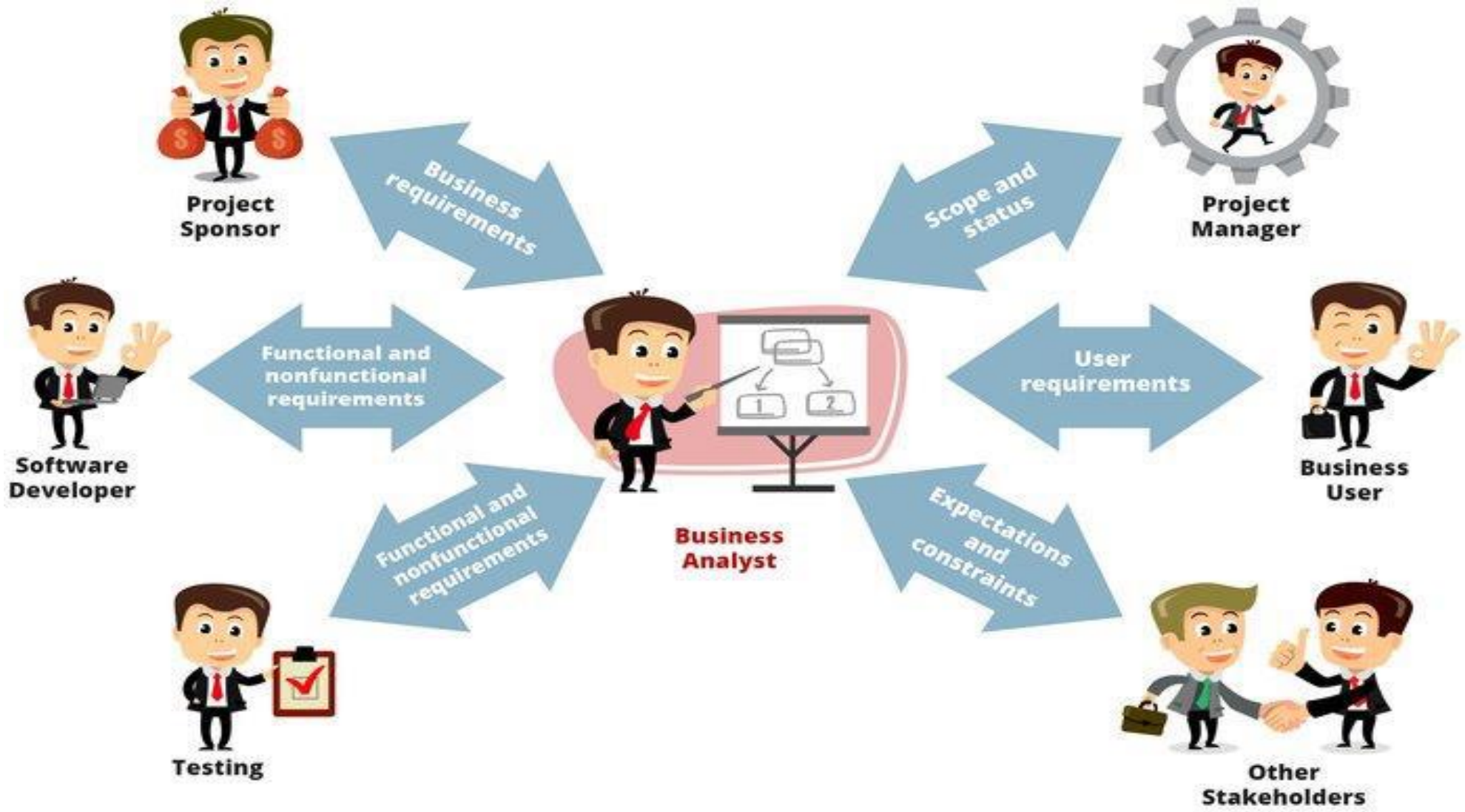
(alınan tum kararların ve gereksinimlerin dokümü)

- **Functional Requirement Document (FRD)** oluşturur.

(yazılımı yapılacak olan bütün maddelerin dokümü)

- Yeteri kadar Functional Requirement toplandıktan sonra **use cases** oluşturur
- Akış şemasını oluşturur

BUSINESS ANALYST (BA)



DEVELOPER (DEV, Yazılımcı)

Developer yazılım programlarının arkasındaki yaratıcı beyindir ve bu programları oluşturmak veya bir ekip tarafından oluşturulmalarını denetlemek için teknik becerilere sahip olan kişidir.

Sorumluluklar:

•Kendilerine aktarılan software requirement dokümanını toparlar ve gereken application ve programın oluşumunu sağlar.

•Beklentileri ve gereksinimleri (costumer requirement) karşılayacak yüksek kalitede (High Quality) code yazarlar.

•Software dokümanını oluşturur ve önceki dökumanları günceller.

QUALITY ANALYST (QA, Tester)

Müşteri ve kullanıcı memnuniyetini göz önünde bulundurarak analiz ve test aşamalarında gerekli düzenlemeleri yapan kişidir.

- Son kullanıcıya bırakılmadan önce analiz ve testlerdeki tüm hataların düzeltilmesini sağlayarak hatasız ürünler sunmak.
- Her hangi bir organizasyonun ürünlerini ve hizmetini beklenen kalite standartlarını karşılayacak şekilde oluşturulmasını sağlar.
- Oluşturlulan application ın istenilen plan çerçevesinde yapılmasını sağlar.
- Application daki hatalar Quality Analyst tarafından bulunmalıdır ki Developer lar bulunan hataların üzerinde calışıp sorun teşkil etmeyecek ürün ortaya koyabilsinler (minimum seviyede bug).
- Testing yapılmasının amacı her hangi bir application da oluşabilecek hataların ortaya cıkarılmasıdır.

Bugün Neler Öğrendik?

SDLC: Yüksek kaliteli (high quality) ve kullanıcı beklentilerini (user expectaion) karşılayan yazılımları tasarlamak, geliştirmek ve test etmek için yazılım endüstrisi tarafından kullanılan bir süreçtir.

SOFTWARE DEVELOPMENT PHASES

(Yazılım Geliştirme Aşamaları)

Planlama (Planning)

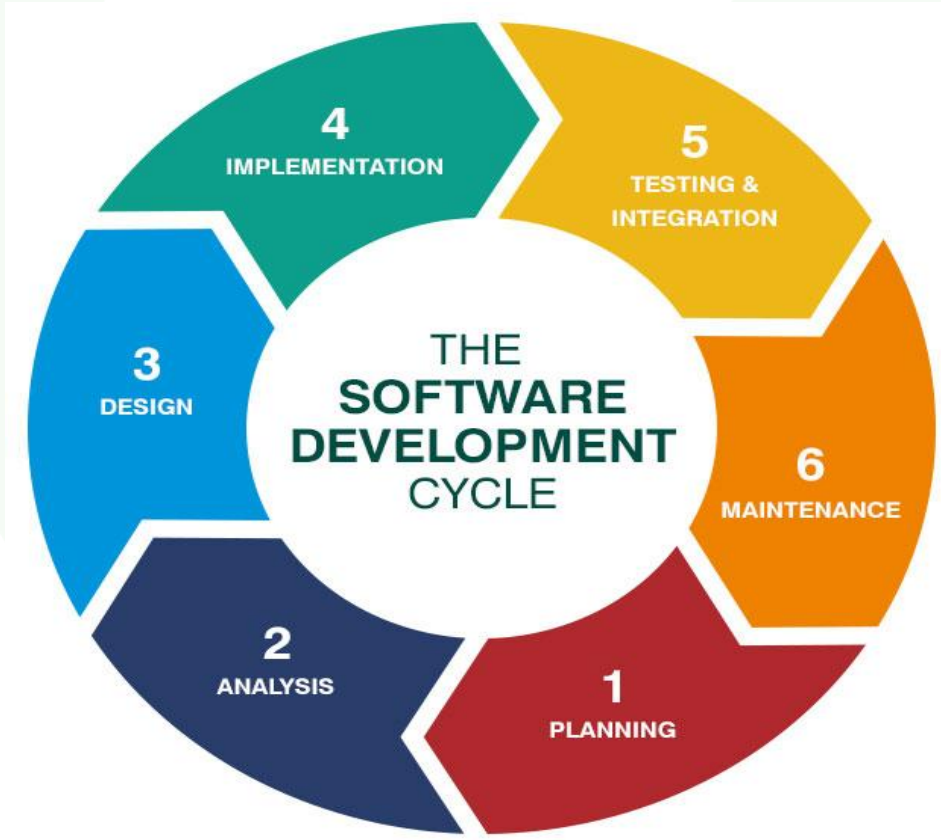
Analiz (Analysis)

Tasarım (Design)

Kodlama (Implementation)

Test (Testing)

Teslim ve Bakım (Maintenance)



Bugün Neler Öğrendik?

SDLC TEAM

1) Project Manager (PM) Proje Yöneticisi
2) Business Analyst (BA) İş Analisti
3) Developer (Dev) Yazılımcı
4) Quality Analyst (QA) Kalite Analisti (Tester)

Business Requirement Document (BRD): Alınan tüm kararların ve gereksinimlerin dökümü

Functional Requirement Document (FRD): Yazılımı yapılacak olan bütün maddelerin dökümü

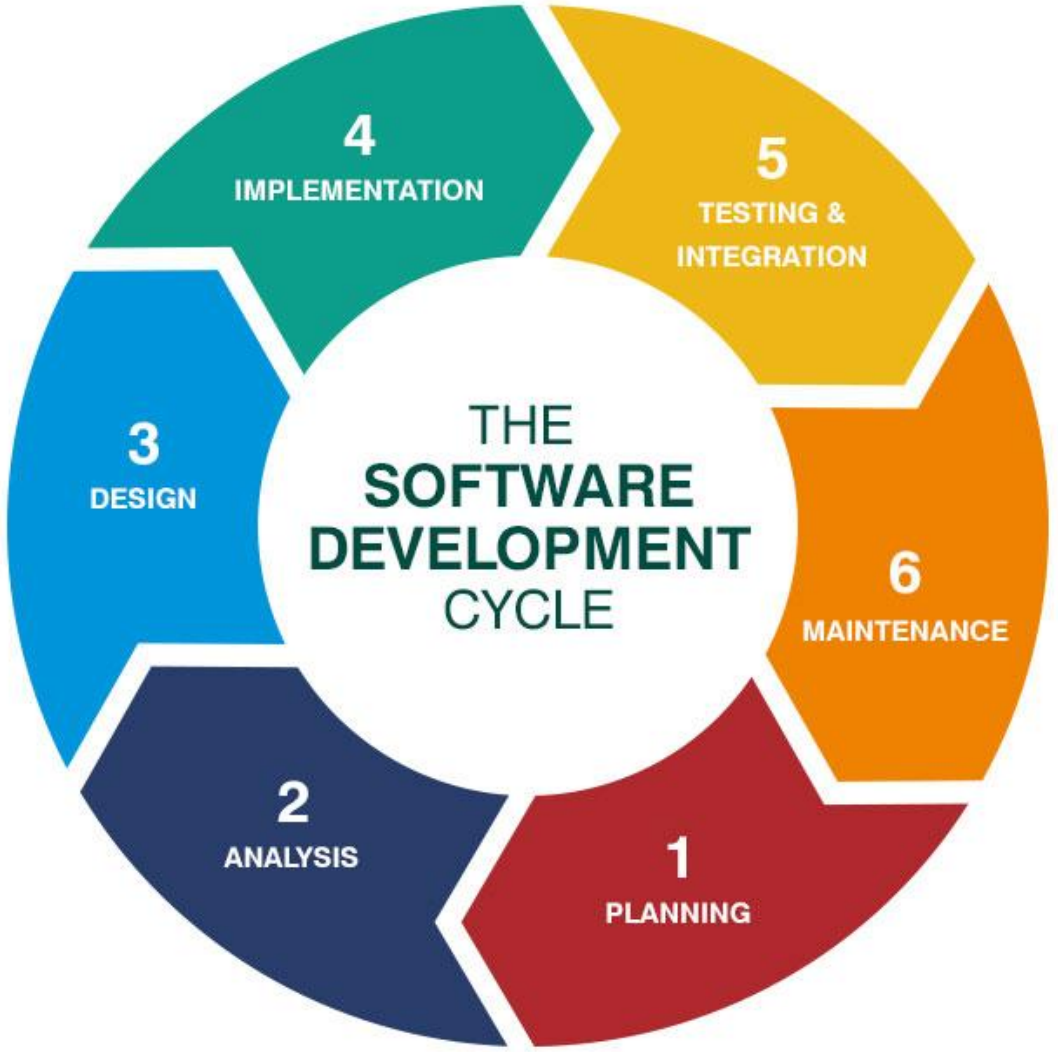
Software Development Life Cycle

(Yazılım Geliştirme Yaşam Döngüsü)

2. Ders
07/05/2022

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

1) Planning and Requirement Analysis
(Planlama ve İhtiyaç Analizi)
2) Defining Requirements
(Gereksinimleri Tanımlama)
3) Designing the product architecture
(Ürün dizaynını tasarlama)
4) Building or Developing the Product
(Ürünü oluşturma veya geliştirme)
5) Testing the Product (Ürünü test etme)
6) Deployment in the Market and Maintenance (Ürünü pazarlama ve bakım)



SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

1) Planning and Requirement Analysis (Planlama ve İhtiyaç Analizi)

- İhtiyaç analizi SDLC'nin en önemli ve temel aşamasıdır.
- Müşteriden gelen fikirler de göz önünde bulundurularak ekibin kıdemli üyeleri (expert) tarafından gerçekleştirilir.

- Bu bilgiler daha sonra temel proje yaklaşımını planlamak için kullanılır.
- Kalite güvence gerekliliklerinin planlanması ve projeyle ilişkili risklerin belirlenmesi de planlama aşamasında yapılır.
- Minimum risklerle projeyi başarıyla uygulamak için izlenebilecek teknik yaklaşımlar planlanır.

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

2) Defining Requirements (Gereksinimleri Tanımlama)

İhtiyaç analizi yapıldıktan sonraki adım, ürün gereksinimlerini açıkça tanımlamak ve belgelendirmektir (dokumante etmek) Stakeholder / işletmeciden onay alınır. Bu proje yaşam döngüsü boyunca tasarlanacak ve geliştirilecek tüm ürün gereksinimlerini içeren

BRD (Business Requirement Document): İş Gereksinimleri Dokümanı iş ihtiyaçlarını ve paydaş gereksinimleri açıklayan bir gereklilik paketidir.

FRD (Functional Requirement Document)

Teknik Islev Ihtiyaclari Dokumani hazirlanir

FRD ve BRD en kucuk User Case lere kadar hazirlanir

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

3) Designing the product architecture (Ürün dizaynını tasarlama)

BRD (Business Requirement Document) Dizaynırların geliştirilecek ürün için en iyi dizaynla ortaya çıkacakları referanstır.
BRD'de belirtilen gereksinimlere dayanarak, ürün mimarisi için genellikle birden fazla tasarım yaklaşımı taslağı olusturulur.

DDS(Design Document Specification): Tasarım spesifikasyonu, bir ürün veya süreçle ilgili noktaların bir listesini sağlayan ayrıntılı bir belgedir.

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

4) Building or Developing the Product (Ürünü oluşturma veya geliştirme)

SDLC'ninbu aşamasında gerçek gelişme başlar ve ürün inşa edilir.

- Yazılımcılar (Developers), kuruluşları tarafından tanımlanan kodlama yönergelerine uymak zorundadır.
- Kodlama için FRD baz alinarak
- Developerlar gereken Funcionality'leri oluştururlar.
- Kodlama için C ++, Java, .Net vs. gibi farklı üst düzey programlama dilleri kullanılır.

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

5) Testing the Product (Ürünü test etme)

Bu aşama, ürün BRD'de tanımlanan kalite standartlarına ulaşıncaya kadar, ürün kusurlarının (bug) rapor edildiği, izlendiği, duzeltigi ve tekrar test edildiği aşamadir.

Ürün iş beklentilerini de karşılamalıdır (requirement specifications)
STLC => Software Testing Life Cycle

Test -> Takip -> Bulunan Hatanin Dev. gönderilmesi -> Raporlama-Duzeltme -> Yeniden Test etme -> Onaylama -> Raporlama

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

6) Deployment in the Market and Maintenance (Ürünü pazarlama ve bakım)

- Ürün test edildikten ve onaylandıktan sonra (hazır olduğunda), resmi olarak uygun görülen şekilde release edilir. (piyasaya sürülür).
- Ürün piyasaya sunulduktan sonra mevcut müşteri tabanı için bakımı yapılır.
- Musteriden (End-User) gelen feedbackler ve Teknolojik Gelismeler ile ihtiyaclar yeniden belirlenir ve dongu yeniden baslatilir

SOFTWARE DEVELOPMENT PHASES (Yazılım Geliştirme Aşamaları)

CTO => Chief Technology Officer
UX => User Experience Designer (Kullanıcı Deneyim Tasarımcısı)
UI => User Interface Designer (Kullanıcı Arayüz Tasarımcısı

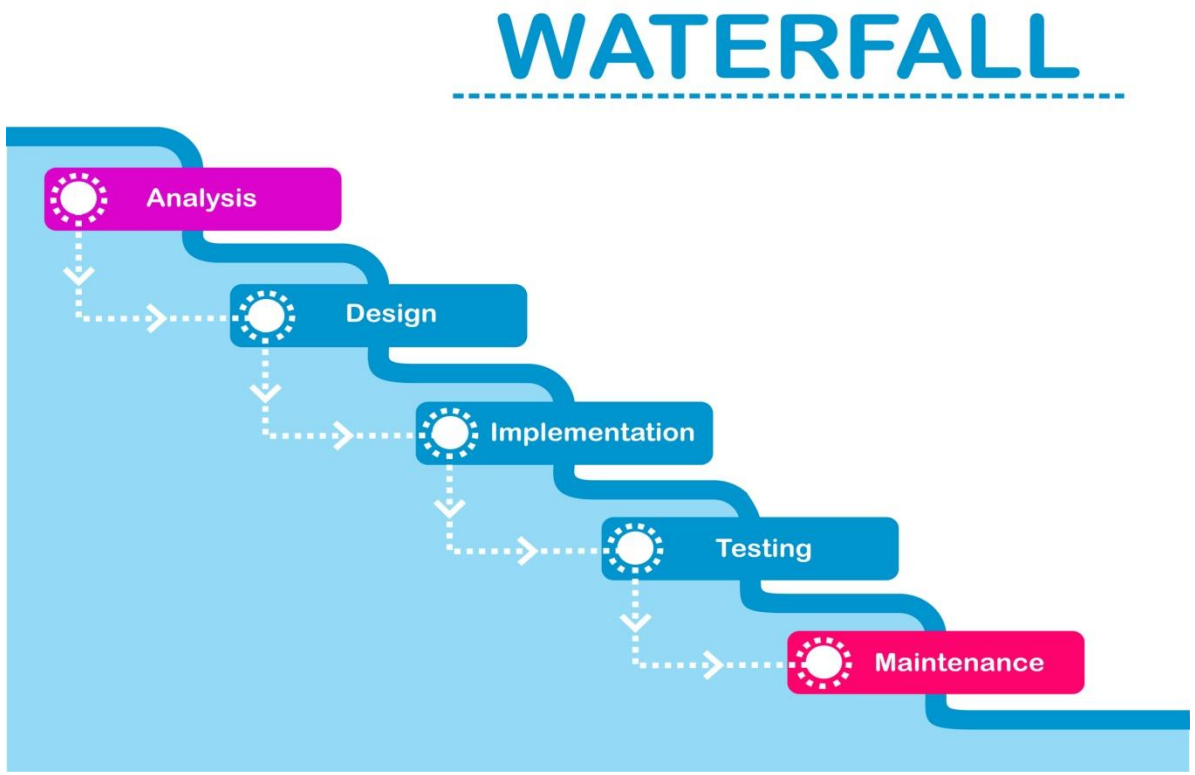
YAZILIM GELİŞTİRMEDE KULLANILAN MODELLER

Yazılım Geliştirme Süreci ile alakalı karşımıza farklı modeller çıkabilir. Kimi şirketler belli bir yönteme bağlı kalmaksızın kendi kültürlerini oluşturmuş olabilirler.

Temelde kullanılan iki modeli inceleyeceğiz

1) WATERFALL MODEL(Şelale Modeli)
2) AGILE METHODOLGY (Çevik Metodoloji)

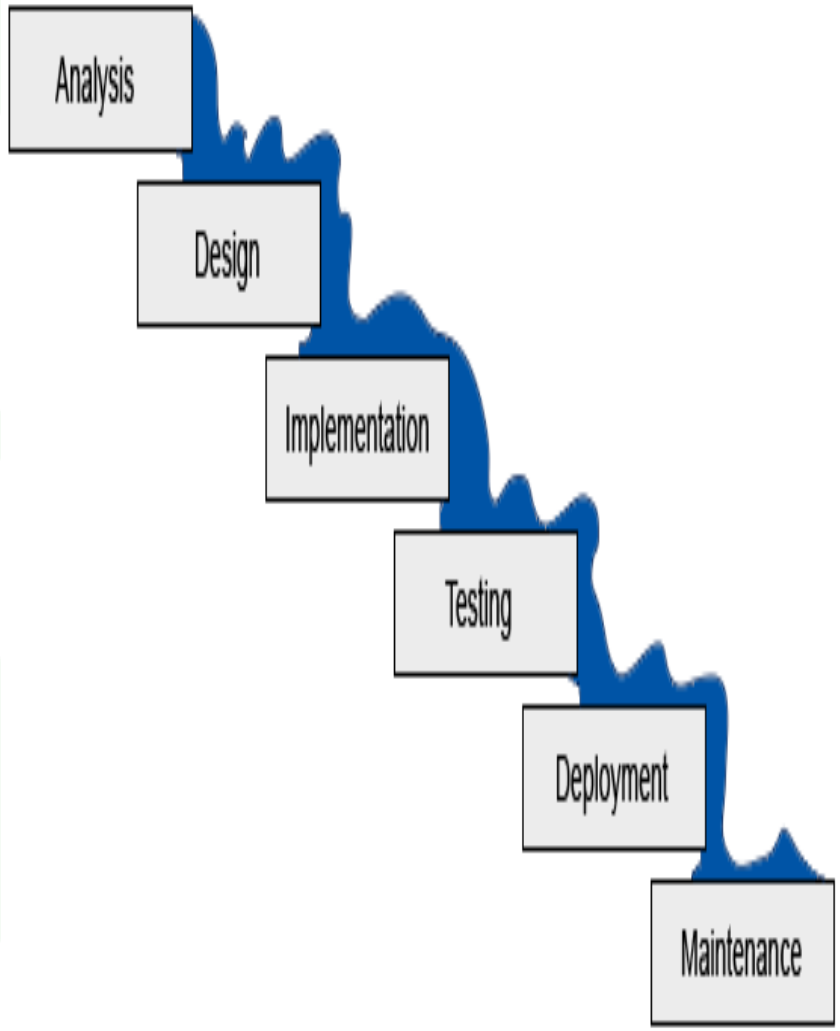
WATERFALL MODEL(Şelale Modeli)



- 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

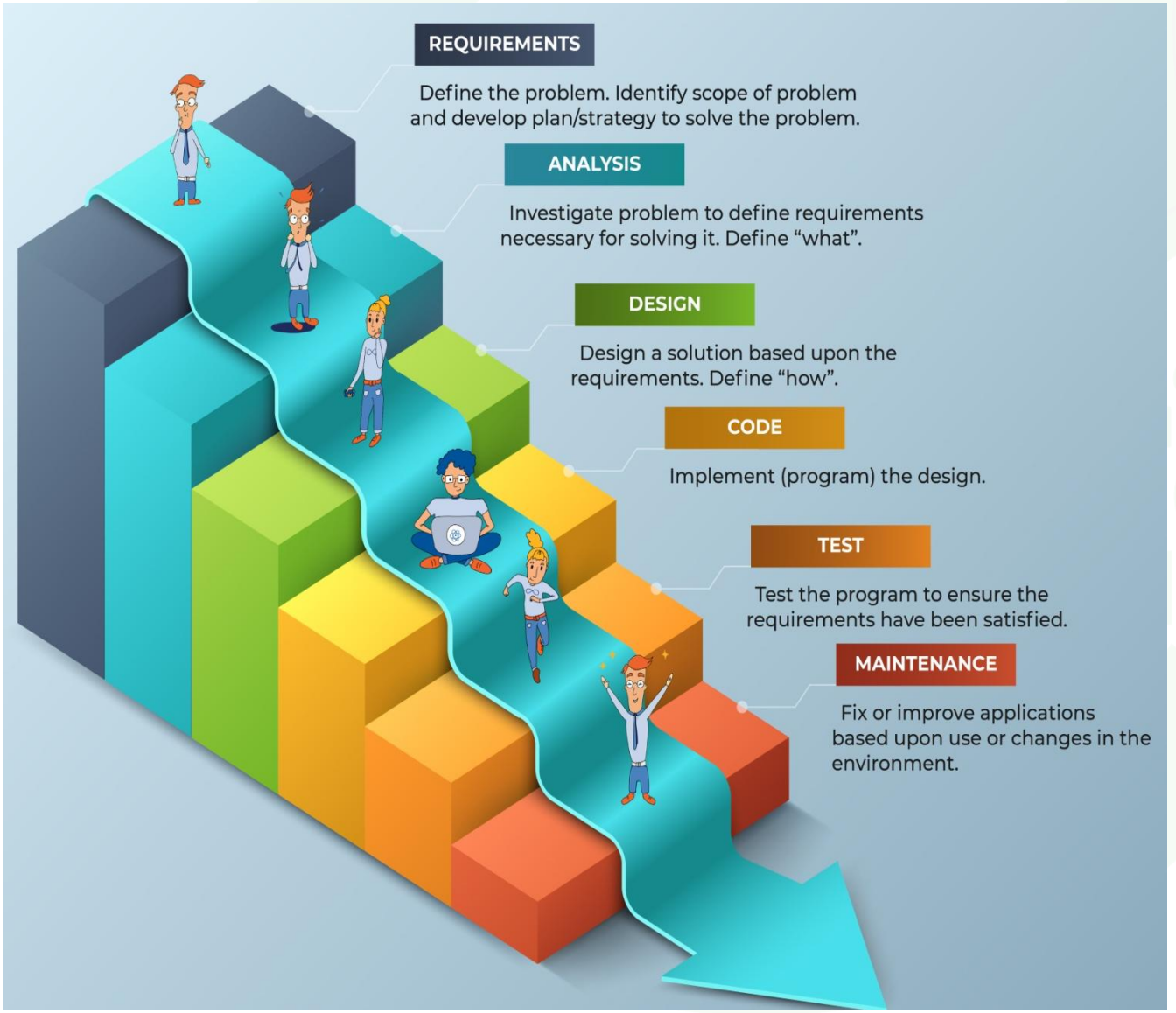
- Kullanımı ve yönetimi kolaydır.
- Gereksinimler iyi anlaşılır.
- Proje bilgisini aktarmak daha kolaydır.
- Küçük projeler için daha iyidir
- Görevler mümkün olduğunca sabit kalır
- Kapsamlı dökümanlar oluşturulur.

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)

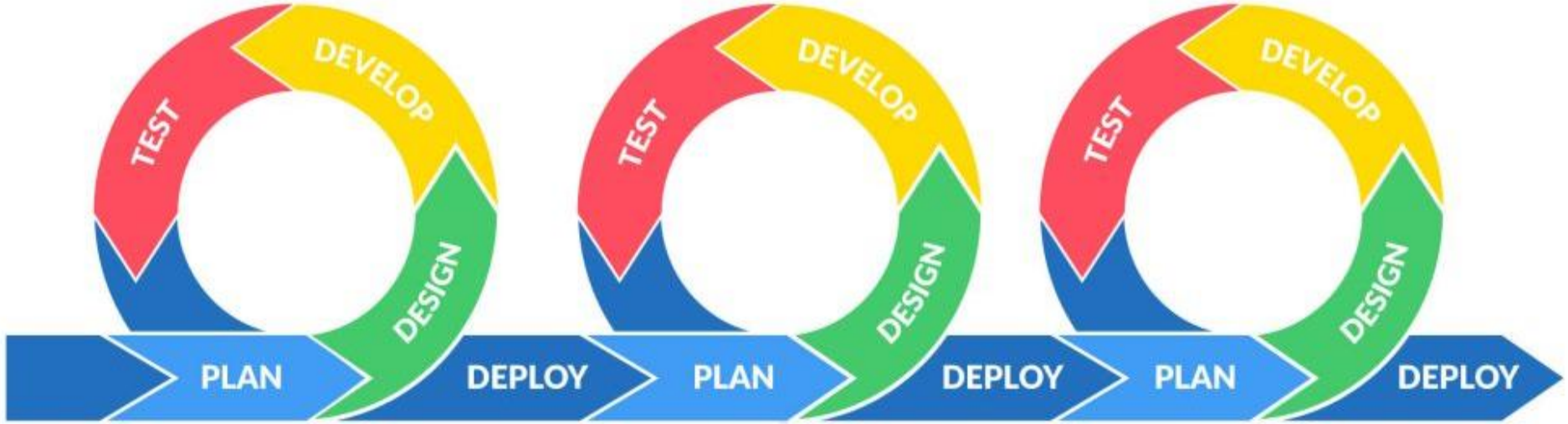
Şelale modelinde analiz ve tasarım aşamaları oldukça detaylı yapıldığından, bu adımlar uzun sürmektedir. Ancak, analiz ve tasarım aşamalarında gereksinimlerin ve tasarımın net bir şekilde ortaya konulmasından dolayı, kodlama ve test aşamaları çok kısa sürmektedir. Test aşamasında çıkan hata sayısı azdır.



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.

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 ilerki 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)

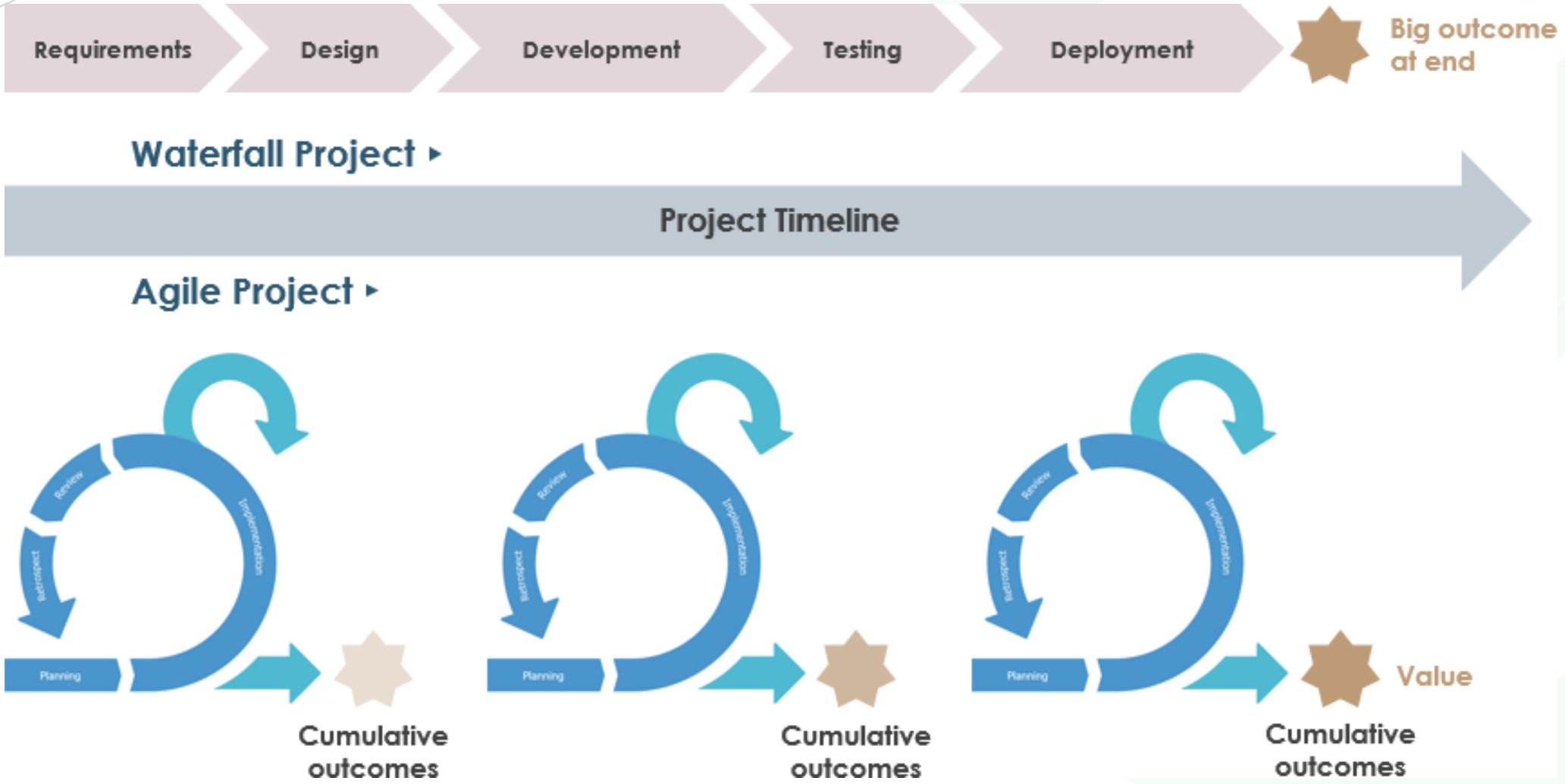
AGILE MANIFESTO

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.

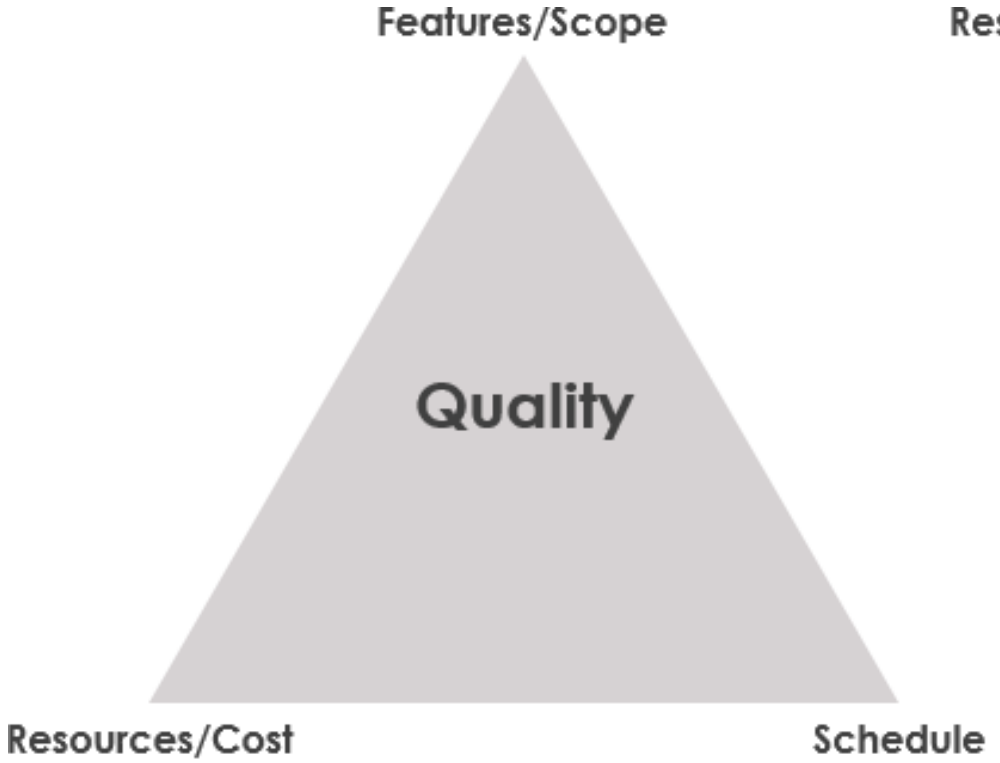
11) En iyi mimariler, gereksinimler ve tasarımlar kendi kendini organize eden ekiplerden ortaya çıkar. En etkin çalışan ekipler kendilerini organize edebilen , bu konuda yetkin ekiplerdir. Ekip kendi çalışma yöntemlerini sorgulamakta ve gerekli değişiklikleri yapmakta özgürdür.

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.

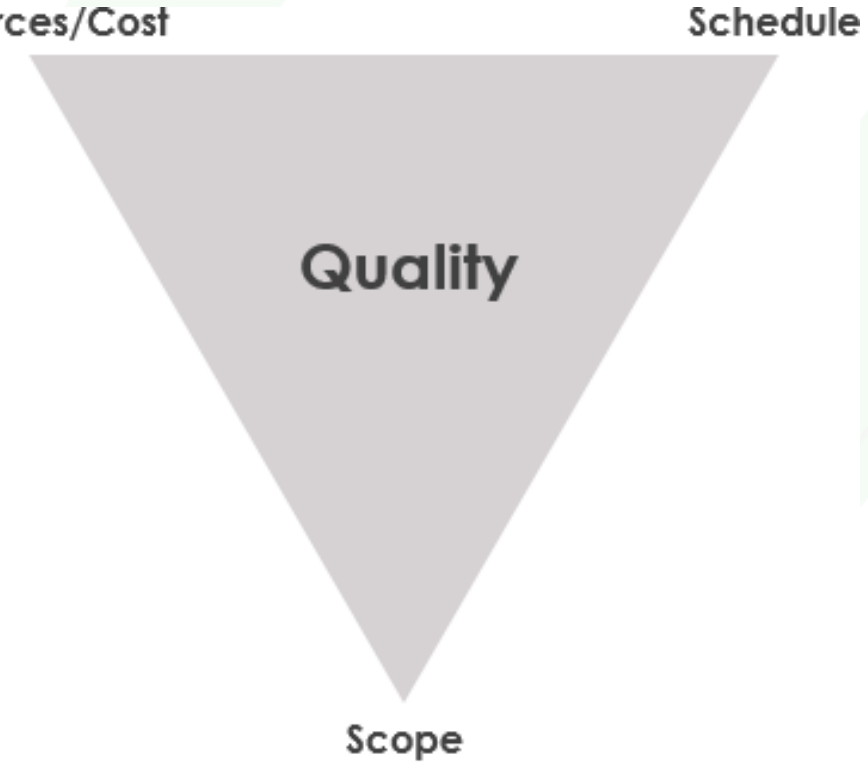
AGILE METHODOLGY VS WATERFALL



AGILE METHODOLGY VS WATERFALL



Waterfall: Scope is fixed, Cost and Schedules are variables

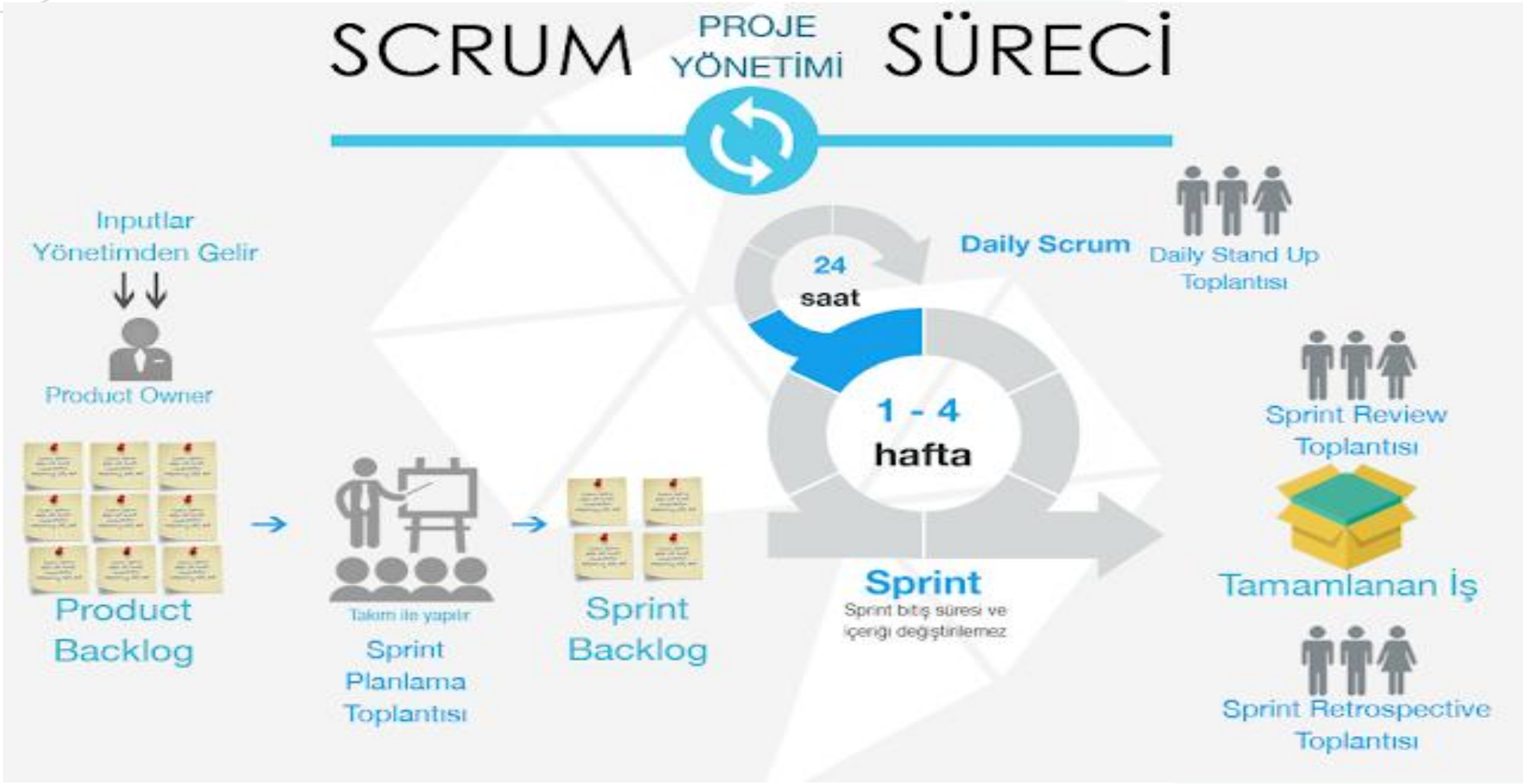


Agile: Cost and Schedules are fixed, Scope is variable

AGILE METHODOLGY VS WATERFALL

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

SCRUM

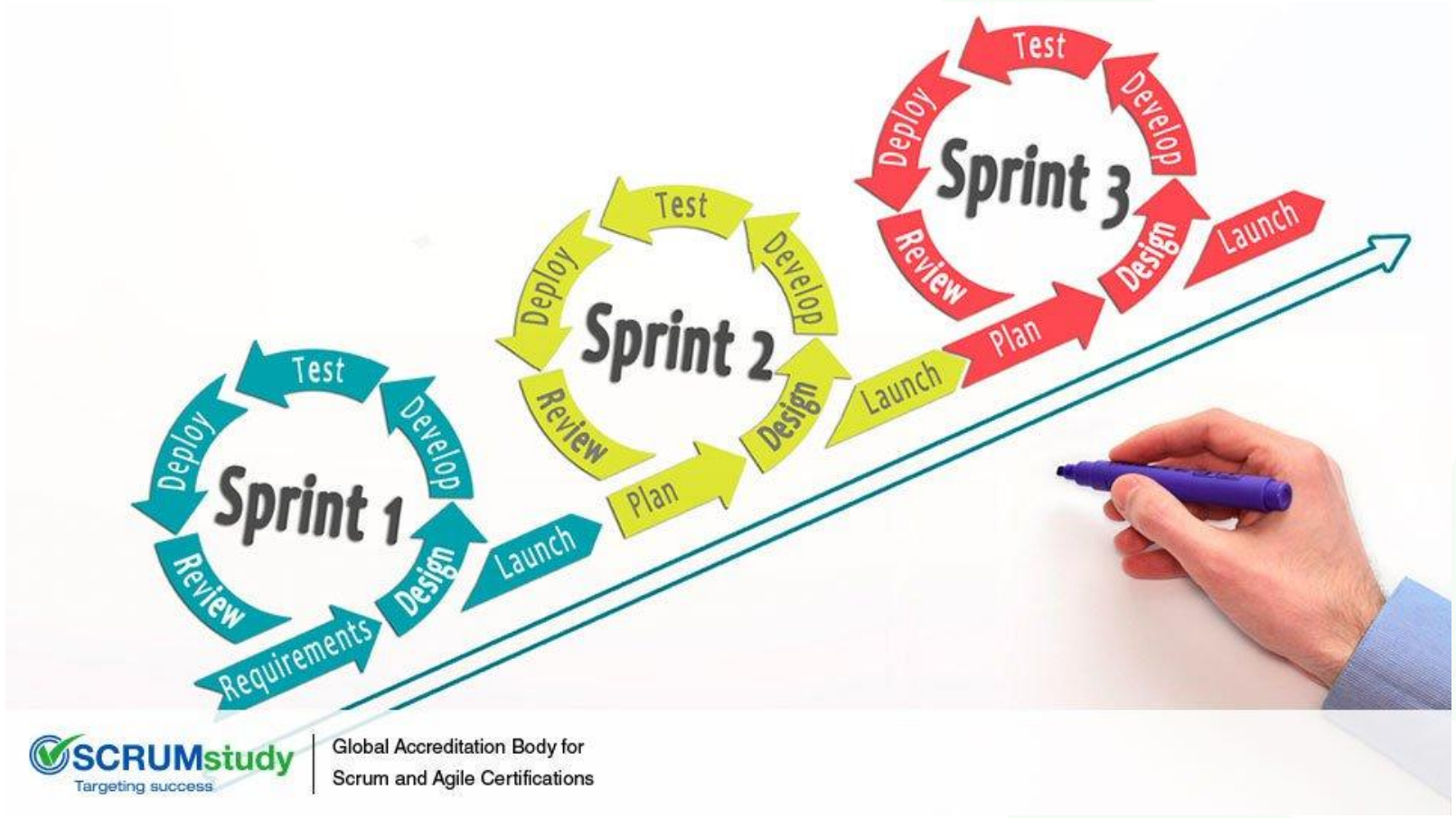

SCRUM PROJE YÖNETİMİ SÜRECİ
Inputlar Yönetimden Gelir
Product Owner
Product Backlog
Takım ile yapılır
Sprint Planlama Toplantısı
Sprint Backlog
24 saat
1 - 4 hafta
Daily Scrum
Daily Stand Up Toplantısı
Sprint
Sprint bitiş süresi ve içeriği değiştirilemez
Sprint Review Toplantısı
Tamamlanan İş
Sprint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

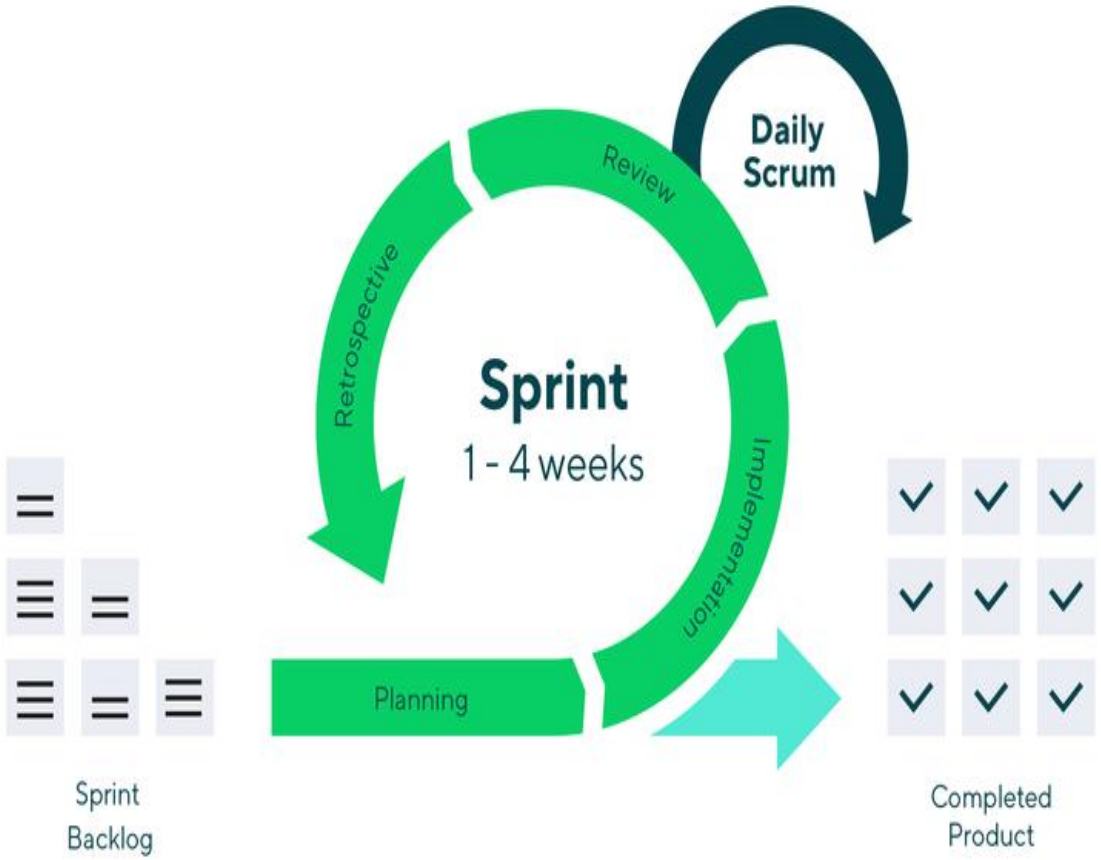
Scrum: Agile proje yönetim metodolojilerinden biridir. Kompleks yazılım süreçlerinin yönetilmesi için kullanılır. Bunu yaparken bütünü parçalayan; tekrara dayalı bir yöntem izler. Düzenli geri bildirim ve planlamalarla hedefe ulaşmayı sağlar. Bu anlamda ihtiyaca yönelik ve esnek bir yapısı vardır. Müşteri 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."

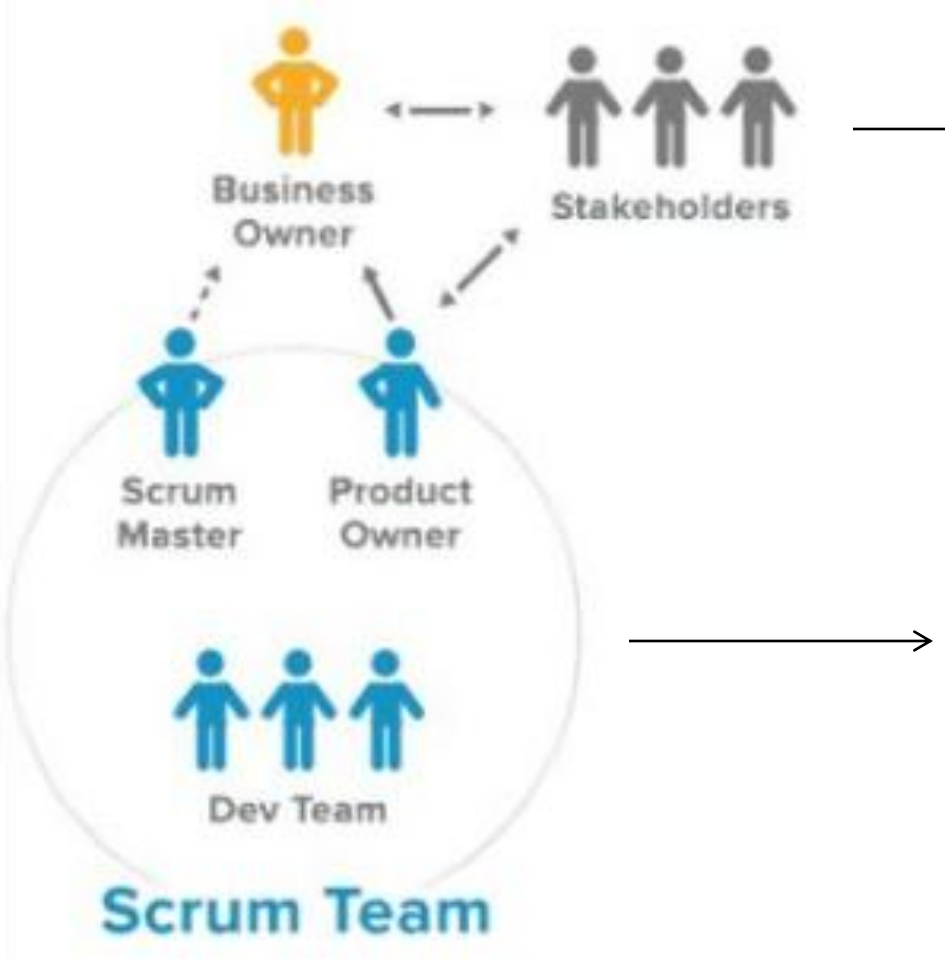
SCRUM



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.

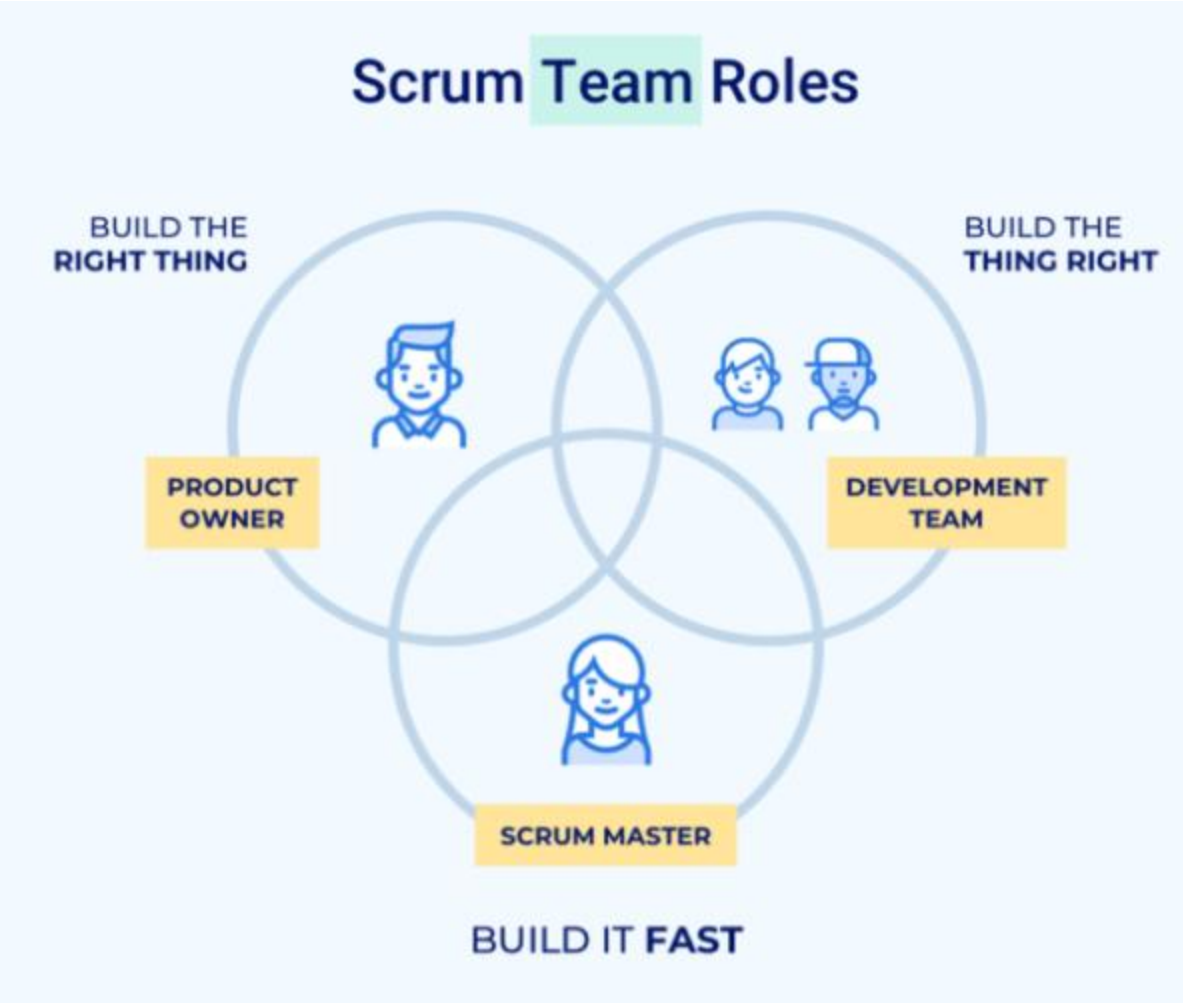
SCRUM TEAM



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.

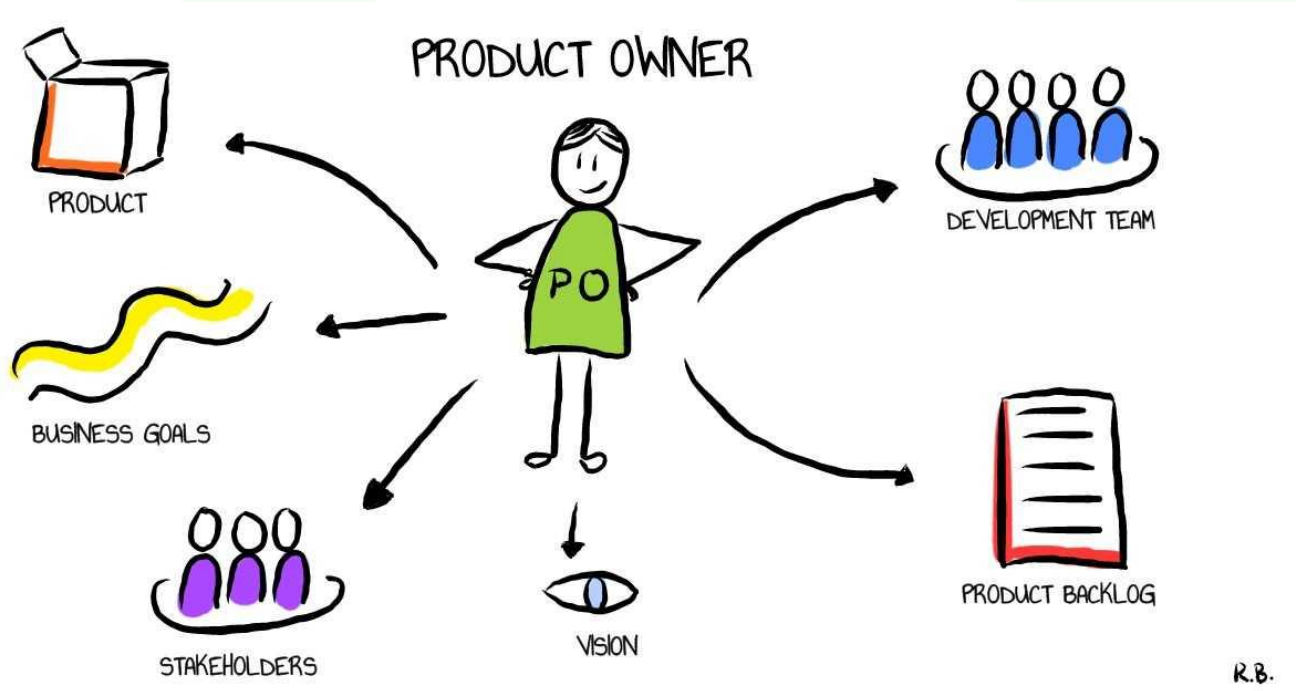
- İş Listesini (Product Backlog) yönetir. Birinci önceliği; iş listesi'ni yönetmektir. Product Backlog'i yönetmek demek, ürünü yönetmek demektir.
- Ürünle ilgili yapılacak bütün geliştirme Product Backlog'da bulunur.
- Product Backlog'un sahibi PO'dur.
- Product Backlog herkes tarafından erişilebilir ve anlaşılabilir olmalıdır.
- Product Owner, Geliştirme Takımı ve Stakeholders arasındaki iletişimi gerçekleştirir.
- Her Sprint'tehangi 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:

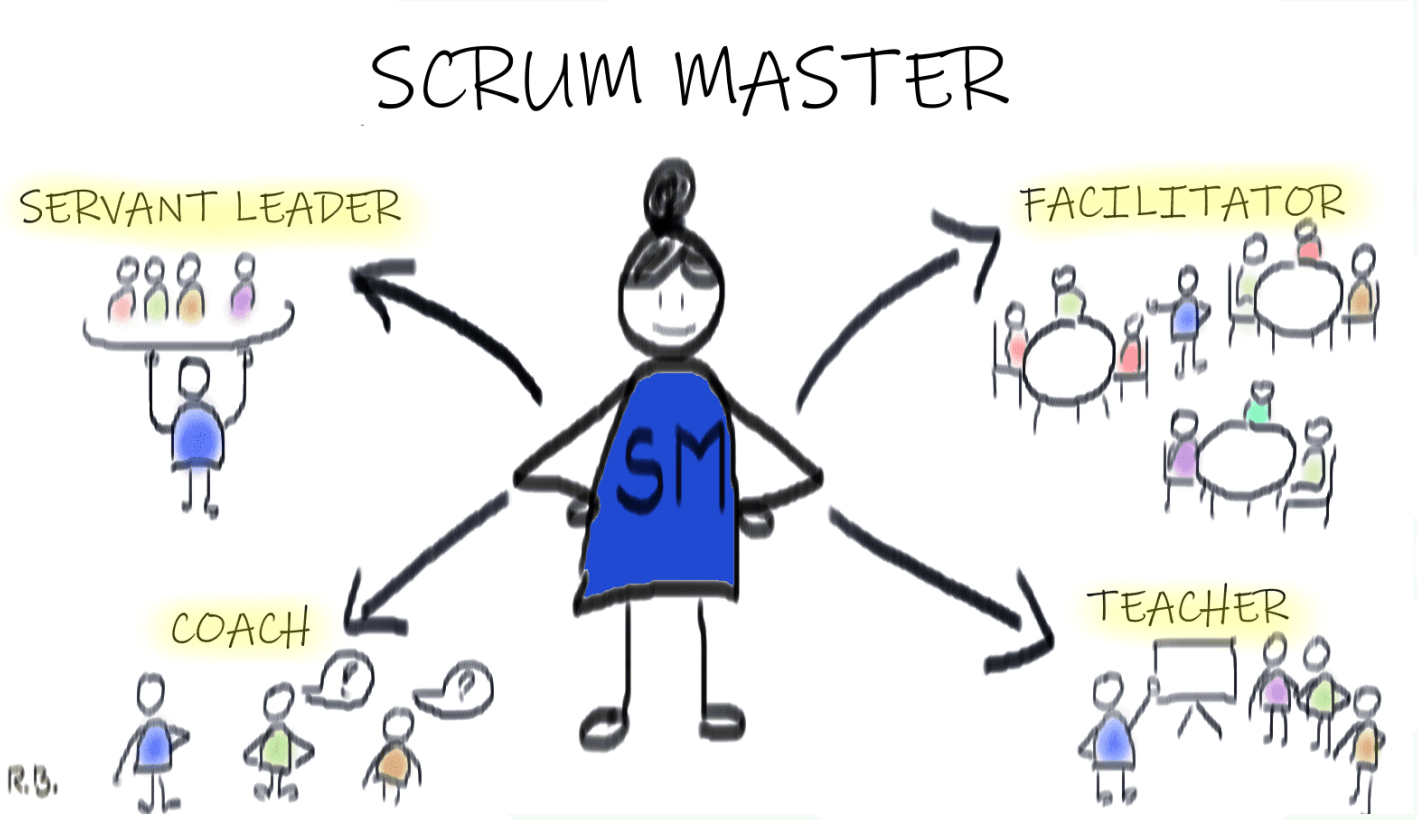
Scrum kurallarını, teorilerini ve pratiklerini iyi bilir ve takımın bu kurallarını uygulamasından sorumlu kişidir. Takımın yöneticisi değildir. Takımı rahatsız eden, verimli çalışmalarını engelleyen durumları ortadan kaldırır.

“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

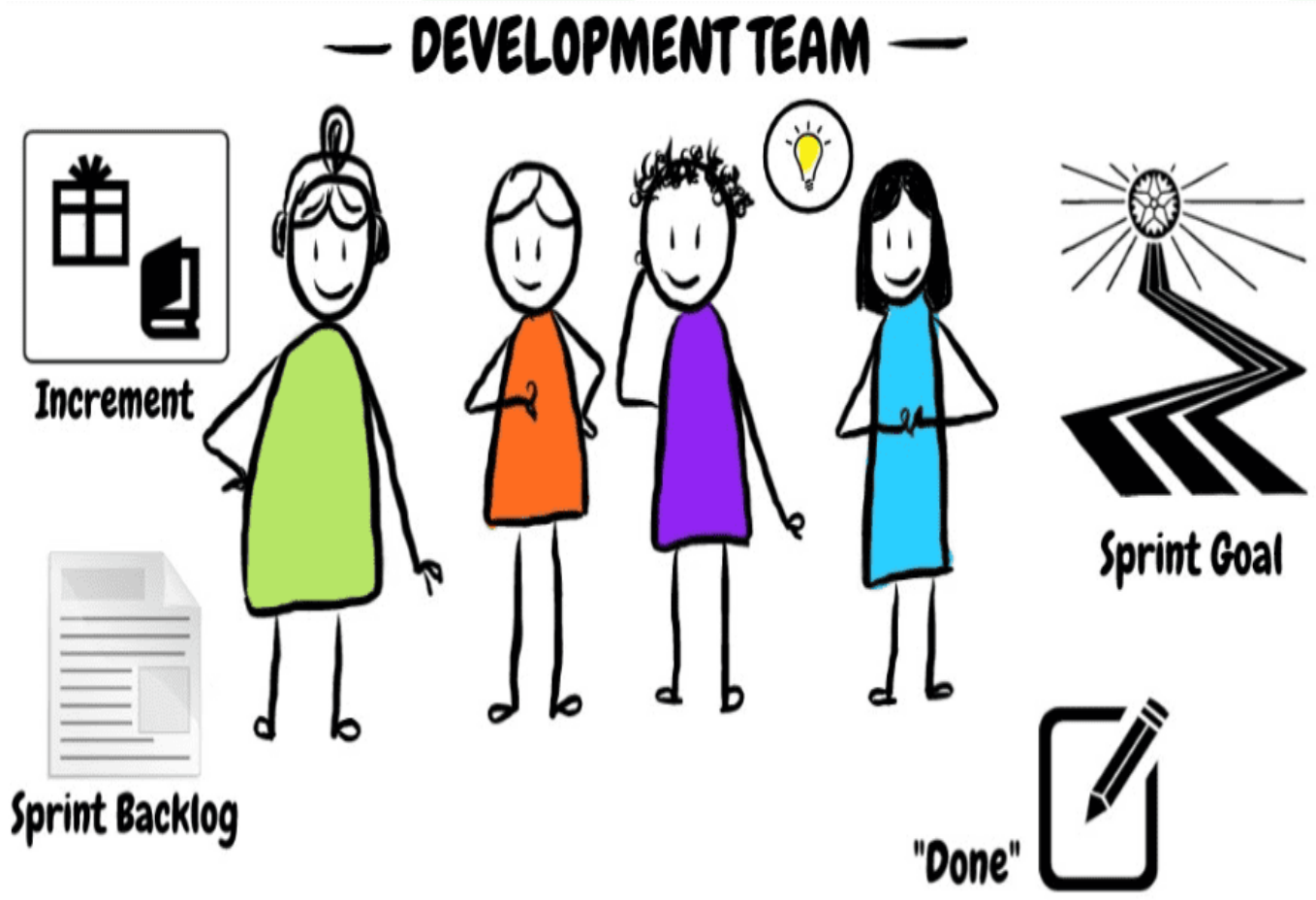
3)Developmet Team:

- 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

Daily Scrum

Daily Stand Up
Toplantısı

Sprint Review
Toplantısı

Tamamlanan İş

Sprint Retrospective
Toplantısı

Sprint
Sprint bitiş süresi ve
içeriği değiştirilemez