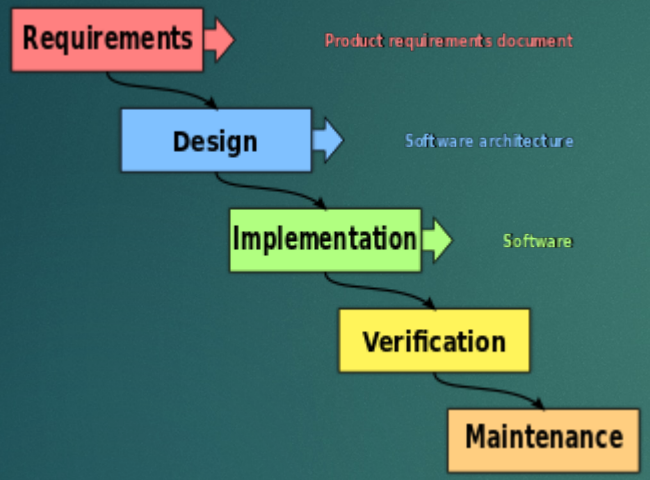
Yazılım Test Mühendisliği, 01.03.2024

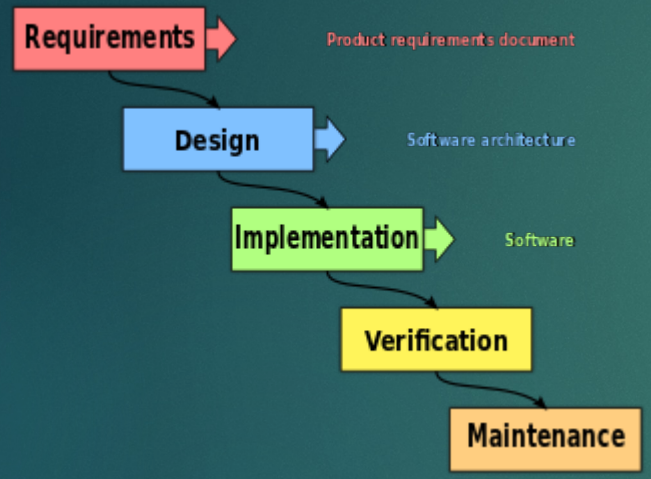
ABDURRAHMAN AKIN

Yazılım Mühendisliği Nedir?



Yazılım mühendisliği, yazılım geliştirme ile ilgilenen bilim dalıdır. Yazılım mühendisliği tanımı ilk olarak 1968 yılında gerçekleştirilen NATO toplantısında Almanya'da gündeme gelmiştir.

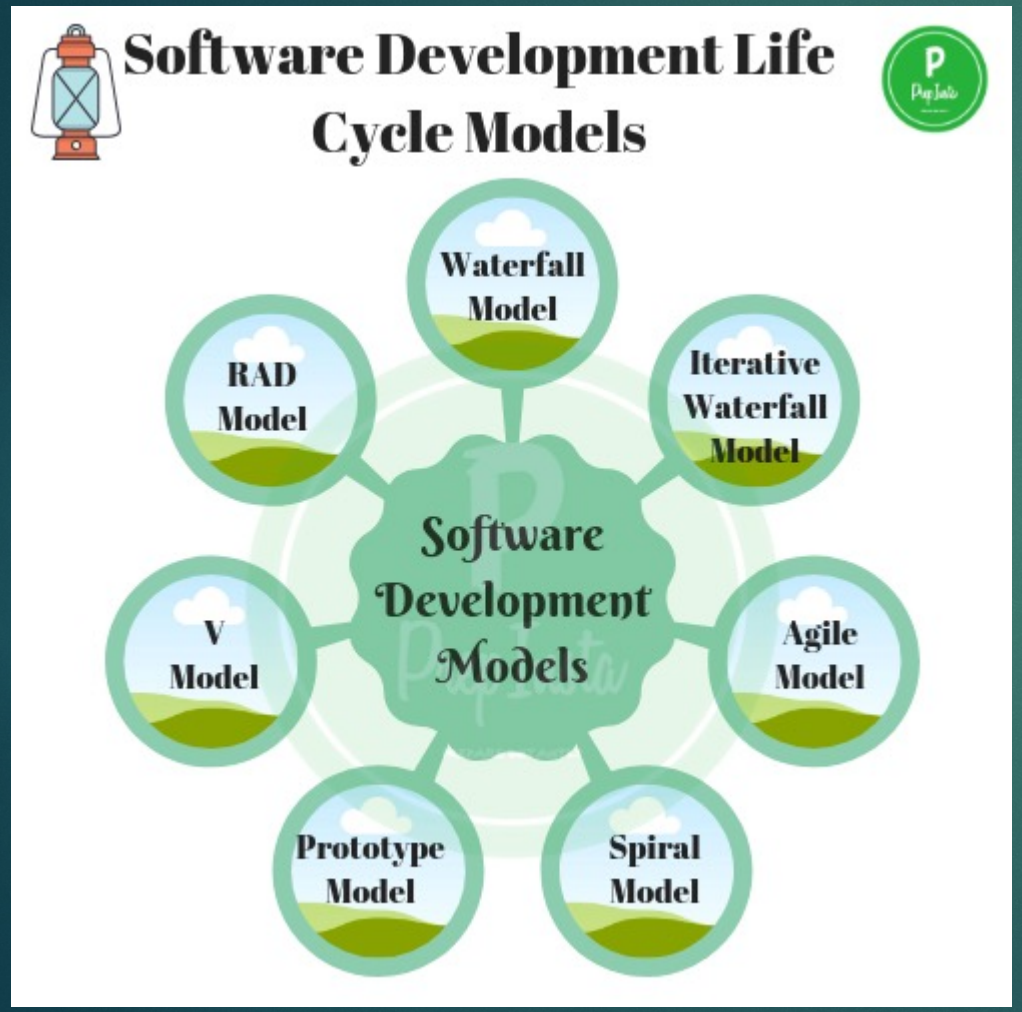
Yazılım Mühendisliği Nedir?



Çekirdek aşamalar:
Planlama
Analiz
Dizayn
Programlama
Test

Destekleyici aşamalar:
Talep yönetimi
Proje yönetimi
Kalite yönetimi
Yapılandırma yönetimi
Yazılım sunumu
Dokümantasyon
Ayrık yapılandırma

Yazılım Geliştirme Modelleri



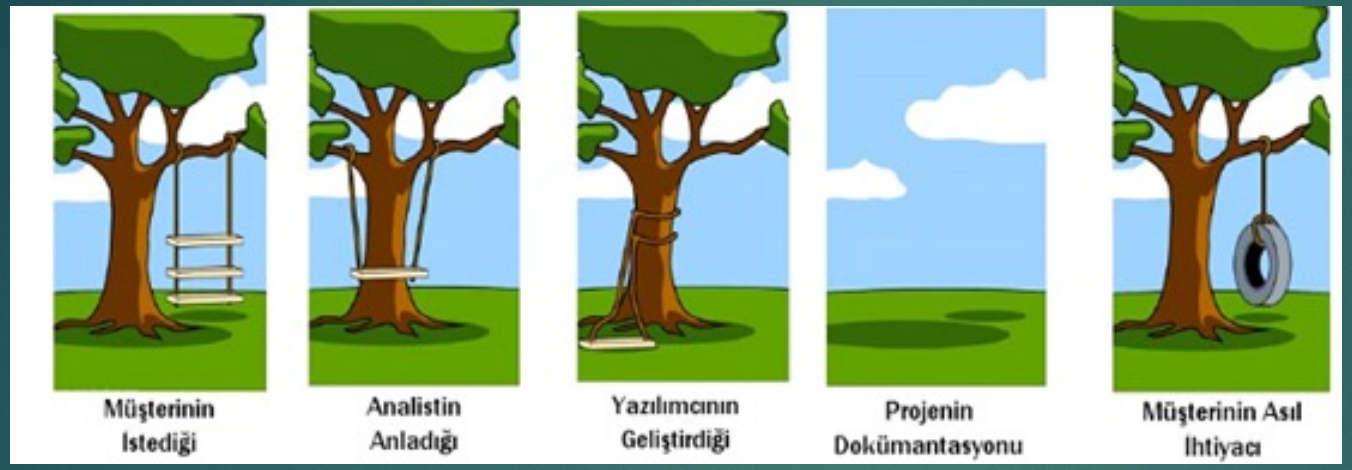
Neden Bu kadar farklı modeller var?

1. Her proje için farklı modeller projenin bağlamına göre avantajlı veya dezavantajlı olabilir

2. Yazılım Meknolojileri geliştikçe modeller de gelişip çeşitlenebiliyor

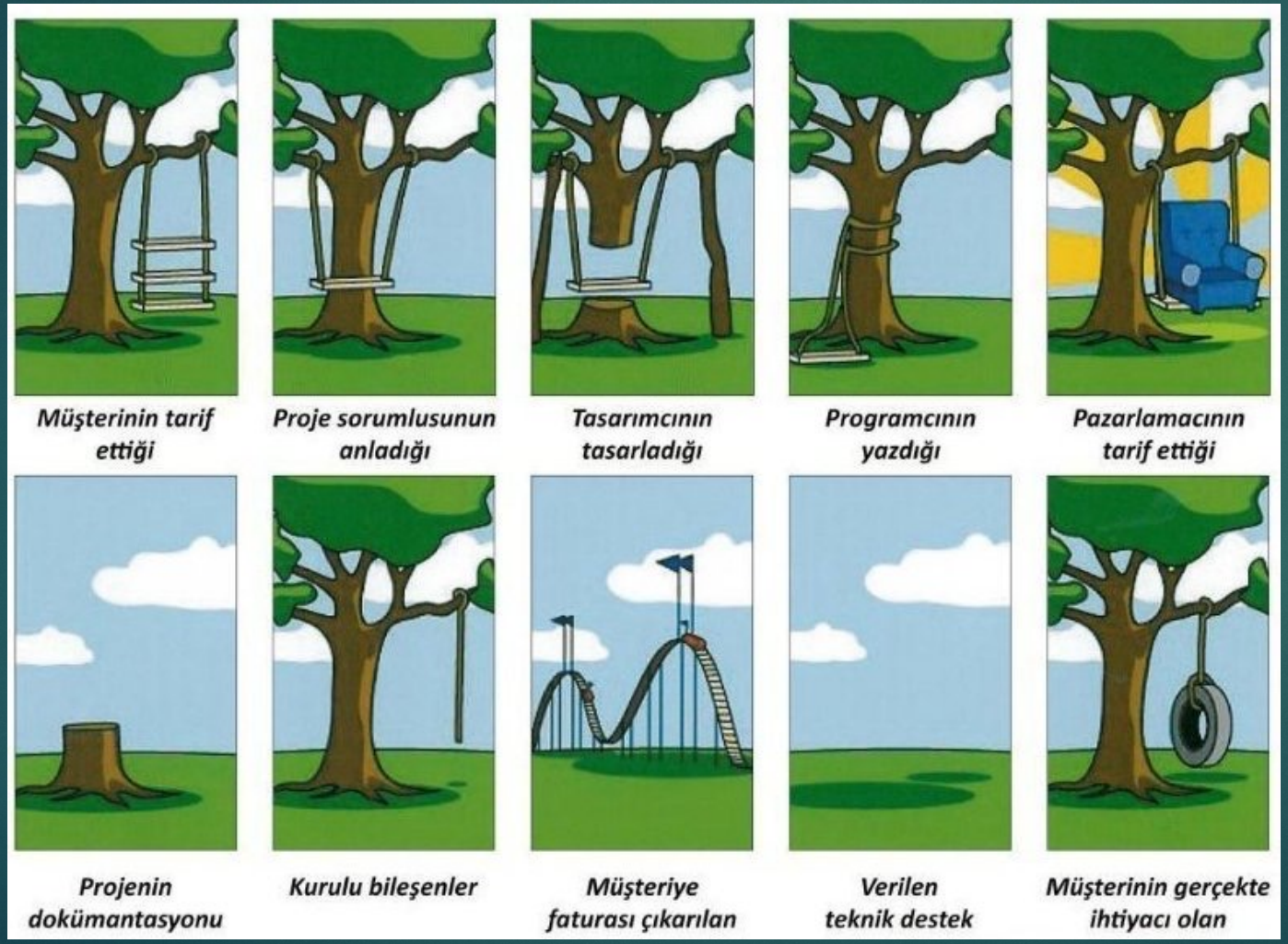
Yazılım Geliştirme Modelleri

Neden Önemli?

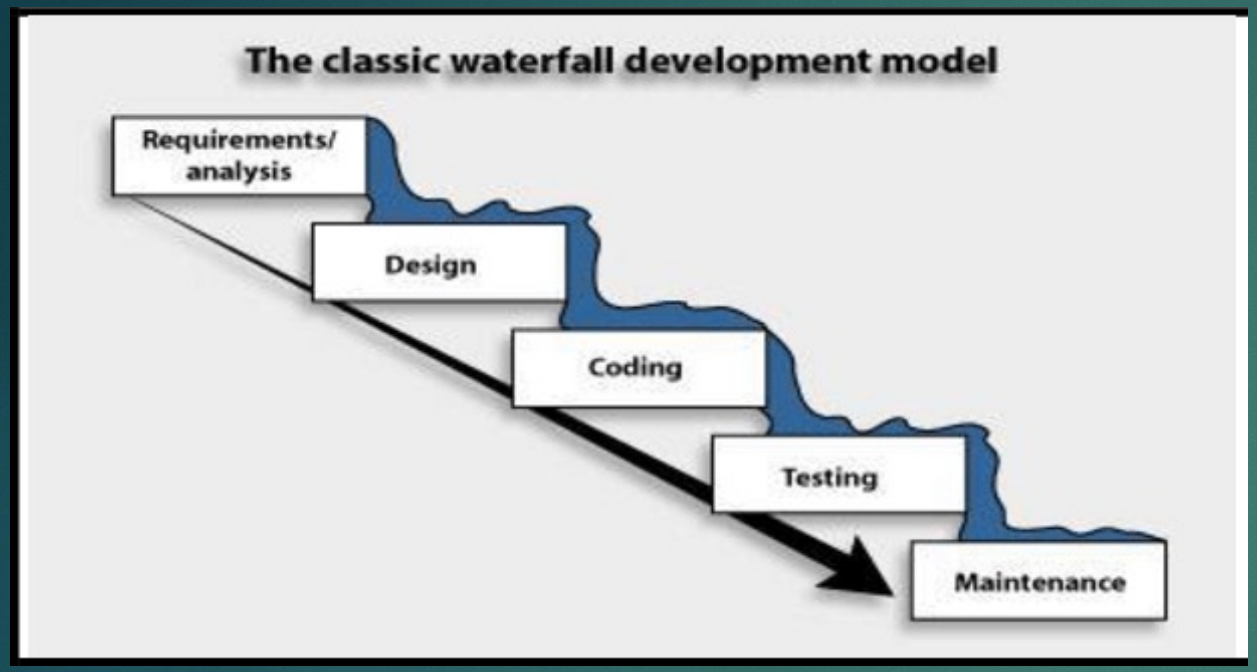


Yazılım Geliştirme Modelleri

Neden Önemli?



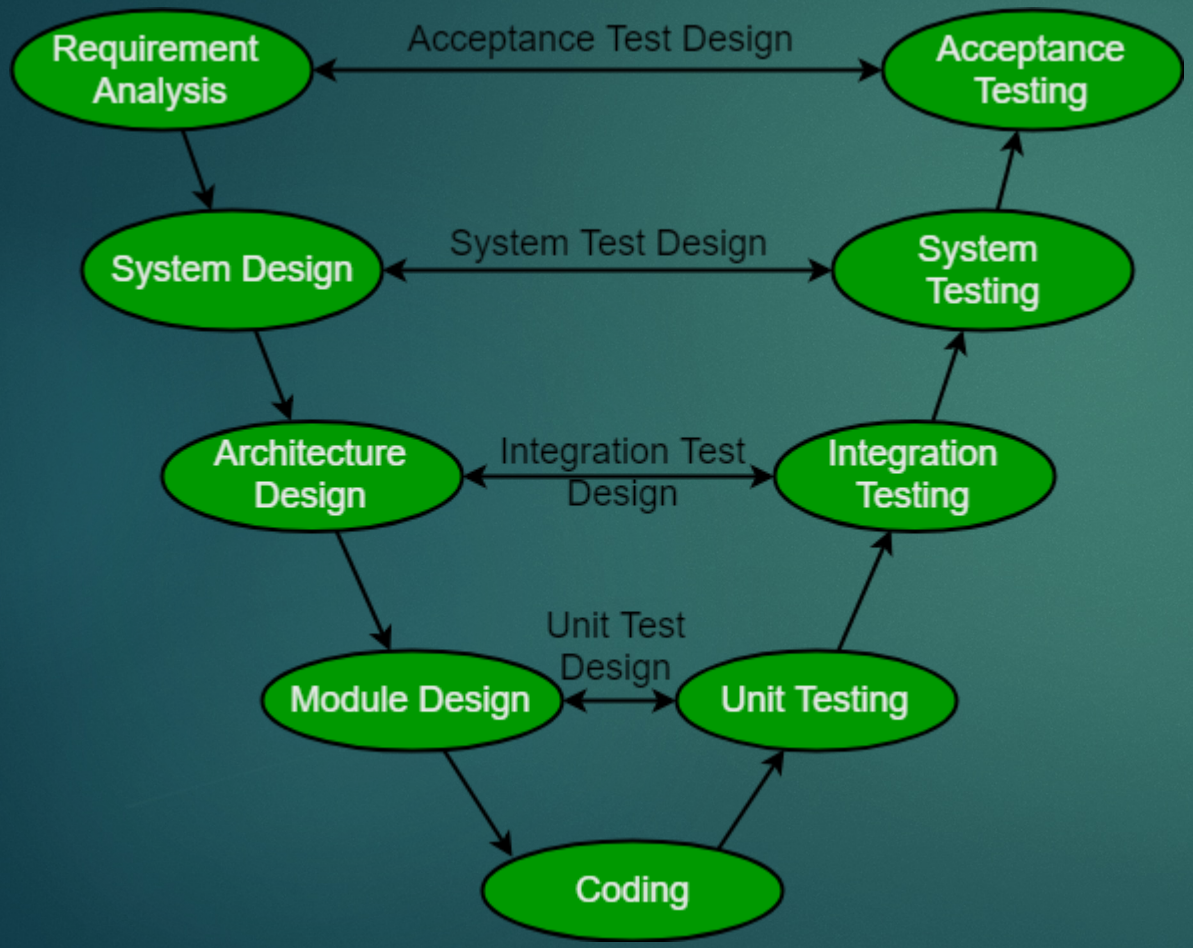
Waterfall Modeli



(+) Kullanımı ve yönetimi kolaydır
(+) Süreci disipline eder
(+) İyi bir dökümantasyon gerektirir

(-) Değişiklikler kolayca yerine getirilemez
(-) Yazılım en sona kadar teslim edilemez
(-) Doğru gereksinimleri toplamak zor olabilir

V Modeli

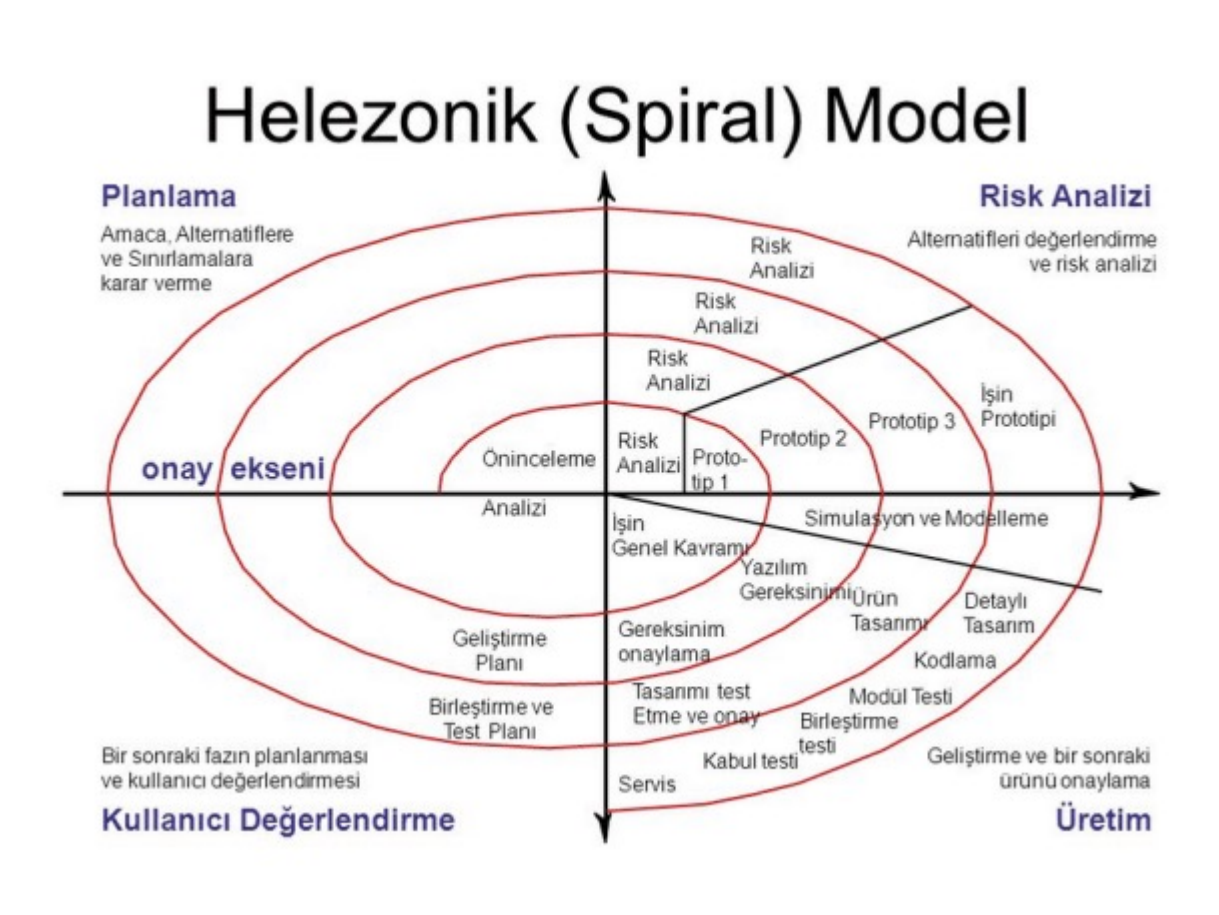


(+) V-modelinde hangi aşamanın ne şekilde test edileceği belirgin bir şekilde gösterilmiştir. Bu sayede yapılanların testi ve doğrulanması planlı bir sistematik disipline taşınır. Böylece hataların daha kolay fark edilmesi ve düzeltilmesi sağlanabilir

(-) Değişiklikler kolayca yerine getirilemez
(-) Yazılım en sona kadar teslim edilemez
(-) Doğru gereksinimleri toplamak zor olabilir

Süreçler IEEE/IEC 12207 ile tanımlanmıştır ve CMMI ile bu süreçlerin olgunluk seviyesi belirlenmektedir.

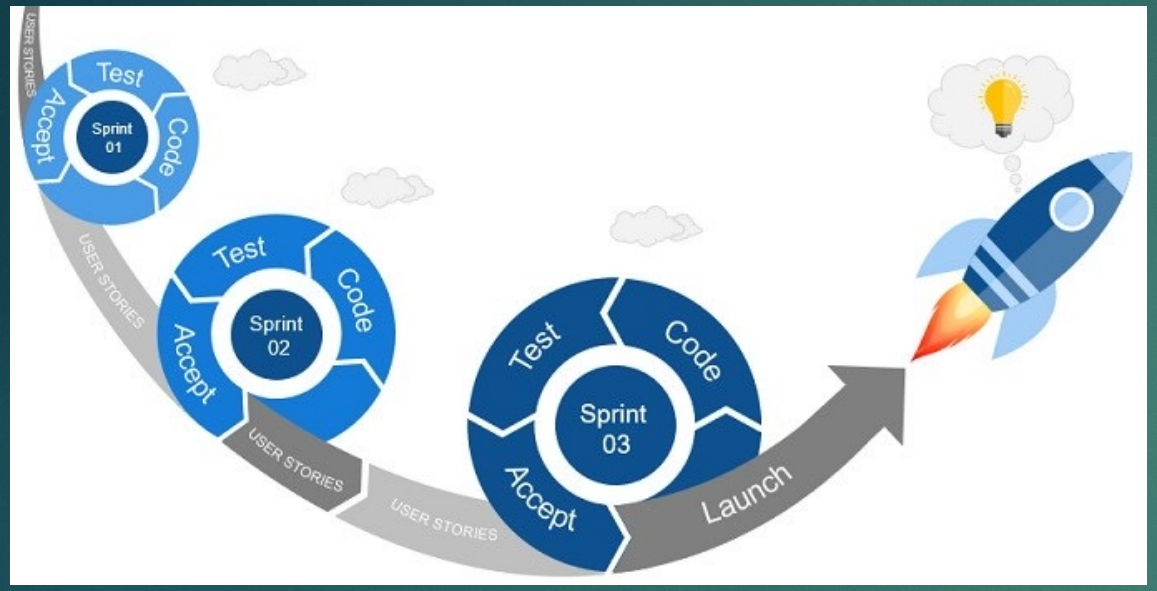
Spiral Model



(+) Kullanıcı sistemi daha erken görebilir.
(+) Parçalara böler, önce risk taşıyanı çözer.
(+) Pek çok yazılım modelini içinde barındırır.
(+) Hataları erken giderme şansı tanır.

(-) Küçük ve düşük riskli projeler için oldukça pahalı bir sistemdir.
(-) Kompleks bir yapıya sahiptir.
(-) Fazla dökümantasyon oluşur.

Agile (Çevik) Modeller



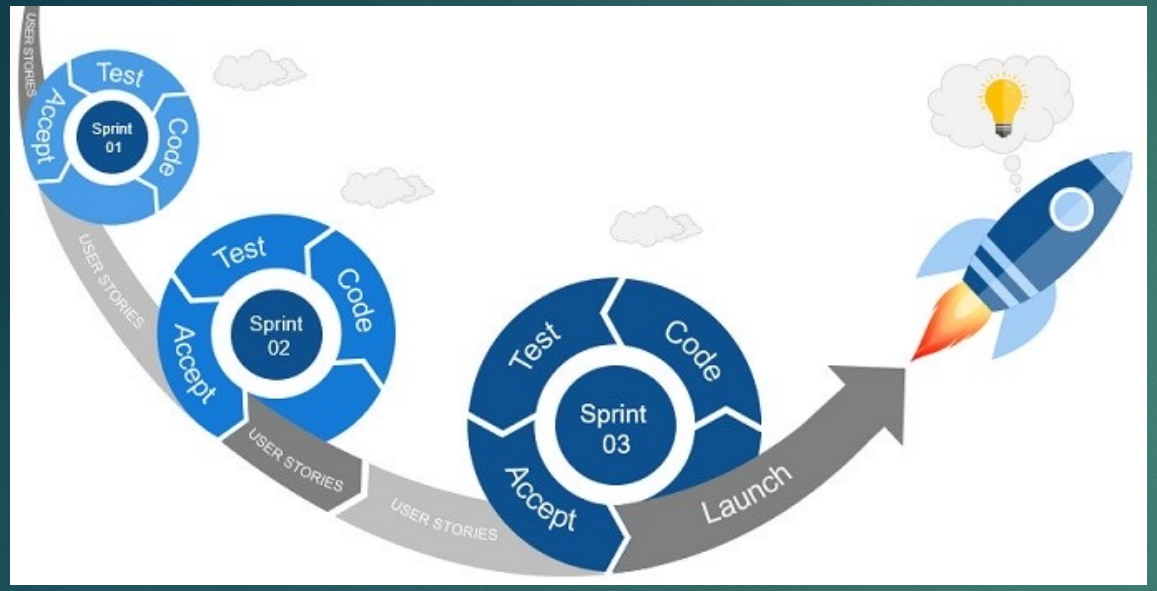
- Scrum
- Test Drive Development
- Extreme Programming
- …

Avantajları:

Kısa planlanmış döngüler, değişen gereksinimler ve ani ortaya çıkabilecek değişiklikler için organizasyonunuza esneklik imkânı tanır. Müşterileriniz ürünü oluşturmak ve çıkan problemler kolayca çözebilmek için aktif bir şekilde geribildirim sağlar.

Ekip katılımına öncelik verilerek kurulan iletişim sayesinde sorunların daha hızlı çözülmesini sağlar. Proje hızla ilerlerken ortaya çıkan her problem bir sonraki döngüde daha iyi bir çözüm için basamak olarak kullanılır.

Agile (Çevik) Modeller



- Scrum
- Test Drive Development
- Extreme Programming
- …

Dezavantajları:

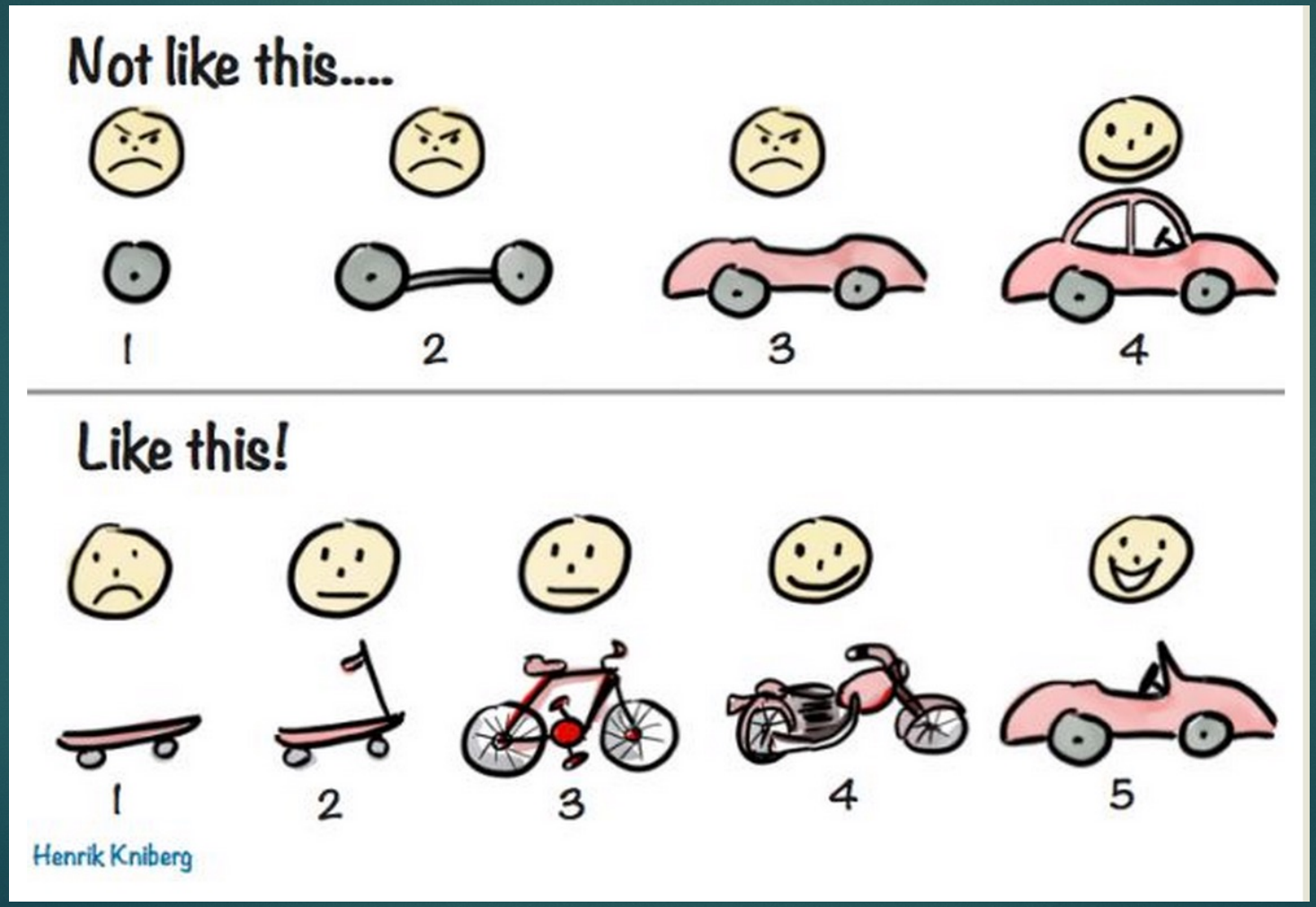
Ürün gereksinimleri sürekli değişebileceğinden maliyetler peşinen tahmin edilemez.

Müşterilerden sürekli geri bildirim almak sizi alaşağı edebilir ya da bazı müşterilerin zamanı ve hiç ilgisi olmayabilir.

Müşteriler geri bildirim vermek istediklerinde akıllarına gelebilecek yeni istekler ek fazları ortaya çıkarır. Bu durum proje maliyetlerini ve süresini uzatabilir.

Müşteri ile iletişimi zor olan büyük Kurumsal yapılarda uygulamak zor olacaktır.

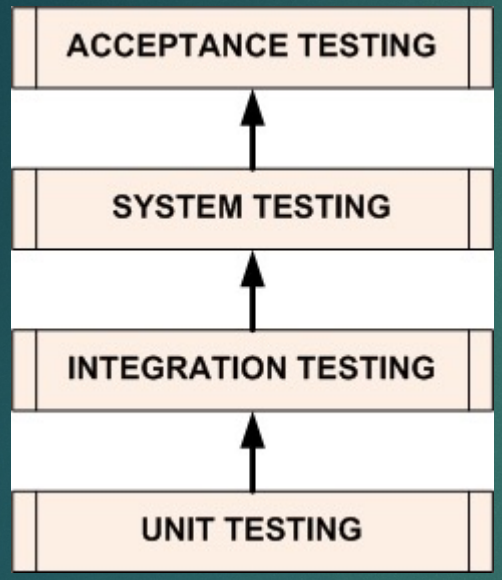
Agile ve Geleneksel Modeller Karşılaştırması



Yazılım Yaşam Döngüsü İçerisinde Test

- Her geliştirme aktivitesi için buna karşılık gelen bir test etme aktivitesi vardır
- Her bir test seviyesinde bu seviyeye özgü test hedefleri bulunur
- Belirli bir test seviyesi için testlerin analizi ve tasarımı karşılık gelen geliştirme akivitesi sırasında başlamalıdır
- Dokümanlar belli bir olgunluğa geldiği anda test uzmanları tarafından gözden geçirilmelidir
- Projenin doğasına veya yazılım mimarisine bağlı olarak test seviyeleri birleştirilebilir veya yeniden düzenlenebilir. Örneğin, ticari bir paket yazılımın (COTS) bir sisteme entegre edilmesi için, sistem seviyesinde entegrasyon testi (örn. altyapıya veya diğer sistemlere entegrasyon ya da sistem dağıtımı) ve kabul testi (fonksiyonel ve/veya fonksiyonel olmayan ve kullanıcı ve/veya operasyonel test) gerçekleştirilebilir.

Yazılım Test Seviyeleri



- Birim Testi
- Entegrasyon Testi
- Sistem Testi
- Kabul Testi

Yazılım Test Seviyeleri

Birim Testi

Test esası:

- ✓ Bileşen gereksinimleri
- ✓ Ayrıntılı tasarım
- ✓ Kod

Kullanılan genel test nesneleri:

- ✓ Bileşenler
- ✓ Programlar
- ✓ Veri dönüştürme / taşıma programları
- ✓ Veritabanı modülleri

Yazılım Test Seviyeleri

Entegrasyon Testi

Test esası:

- ✓ Yazılım ve sistem tasarımı
- ✓ Mimari
- ✓ İş akışları
- ✓ Kullanım senaryoları

Kullanılan genel test nesneleri:

- ✓ Alt sistemler
- ✓ Veritabanı uyarlama
- ✓ Altyapı
- ✓ Arayüzler
- ✓ Sistem yapılandırması ve yapılandırma verisi

Yazılım Test Seviyeleri

Sistem Testi

Test esası:

- ✓ Sistem ve yazılım gereksinimleri
- ✓ Kullanım senaryoları
- ✓ Fonksiyonel gereksinimler
- ✓ Risk analizi raporları

Kullanılan genel test nesneleri:

- ✓ Sistem, kullanıcı ve operasyon kılavuzları
- ✓ Sistem yapılandırması

Yazılım Test Seviyeleri

Kabul Testi

Test esası:

- ✓ Kullanıcı gereksinimleri
- ✓ Sistem gereksinimleri
- ✓ Kullanım senaryoları
- ✓ İş süreçleri
- ✓ Risk analizi raporları

Kullanılan genel test nesneleri:

- ✓ İş süreçleri
- ✓ Operasyonel süreçler ve bakım süreçleri
- ✓ Kullanıcı prosedürleri
- ✓ Formlar
- ✓ Raporlar
- ✓ Yapılandırma verisi