YAZILIM GELİŞTİRMEDE ÇEVİK YÖNTEMLER

GİRİŞ

Yazılım Projeleri

- Yazılım hemen hemen tüm iş faaliyetlerinin bir parçası haline gelmiştir. Bu nedenle yeni yazılımlar, yeni iş fırsatları elde edebilmek ve rekabet baskısıyla baş edebilmek için hızlı bir şekilde geliştirilmelidir.
- Hızlı yazılım geliştirme ve teslimi birçok işletme için en kritik gereksinimlerden biri haline gelmiştir.
- Şirketler yazılımı hızlı bir şekilde teslim edebilmek için kaliteden ödün verme ve bazı gereksinimlerden fedakarlık etme noktasına bile gelebilmektedir.
- Hızla değişen bir ortamda yazılım için her şeyiyle tam bir sabit gereksinim listesi elde etmek imkansızdır.

Yazılım Projeleri

- Gereksinimler değişir, çünkü müşteriler yeni geliştirilen sistemin var olan çalışma şekillerini ya da çalışan diğer sistemleri nasıl etkileyebileceğini ya da sistem dahilinde hangi kullanıcı işlemlerinin otomatikleştirilmesi gerektiğini yeterince hayal edemez.
- Gerçek gereksinimler sadece sistem teslim edildikten ve kullanıcılar sistemi kullandıktan sonra netleşir. O zaman bile dış faktörler gereksinim değişikliğine neden olabilirler.
- Gereksinimlerin belirlenmesi, analiz, tasarım, gerçekleştirim ve bakım aşamalarını içeren plan güdümlü yazılım geliştirme süreçleri hızlı yazılım geliştirme ile örtüşmez.

Yazılım Geliştirme Süreç Modelleri

- Yazılım geliştirme süreç modelleri, yazılım yaşam döngüsünde belirtilen süreçlerin hangi düzen ya da sırada, nasıl uygulanacağını tanımlar.
- Yazılım geliştirmenin, bahsedilen zorluklarıyla baş edebilmek için, geliştirmeyi sistematik hale getirmeyi hedefleyen çeşitli süreç modelleri ortaya çıkmıştır.
- Bu modellerin temel hedefi; proje başarısı için, yazılım geliştirme yaşam döngüsü boyunca izlenmesi önerilen mühendislik süreçlerini tanımlamaktır.
- Modellerin ortaya çıkmasında, ilgili dönemin donanım ve yazılım teknolojileri ile sektör ihtiyaçları önemli rol oynamıştır.
- Yazılım geliştirme süreç modelleri, süreçlerin içsel ayrıntıları ya da süreçler arası ilişkilerle ilgilenmez.

Yazılım Süreç Modelleri Neden Önemli?

- Endüstri kaliteye önem vermektedir. (performans, üretkenlik, vb.)
- Yazılım projelerinin kalitesi ve bütçesi büyük ölçüde seçilen modele bağlıdır.
- Yazılım geliştirmenin karmaşık ve zorlu süreçlerini iyileştirmeyi hedefler.

Yazılım Süreç Modelleri

Düzenleyici Süreç Modelleri

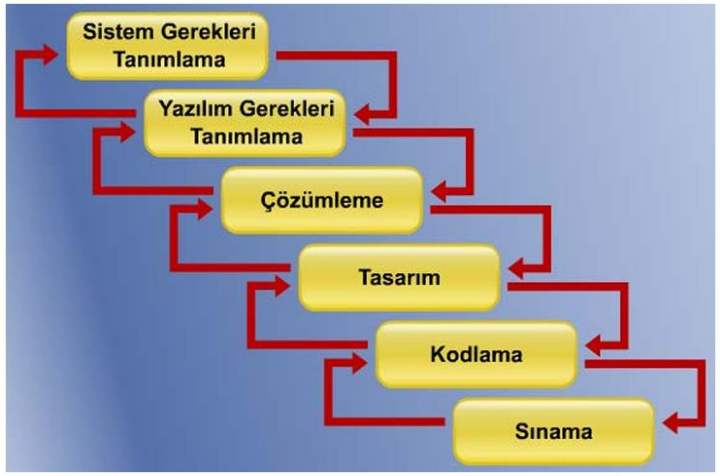
- Kodla ve Düzelt
- Gelişigüzel Model
- Barok Modeli
- Çağlayan/Şelale Modeli
- V Modeli
- Prototipleme
- Helezonik Model
- Evrimsel Geliştirme Modeli
- Artırımsal Geliştirme Modeli
- Araştırma Tabanlı Model
- Formal Sistem Geliştirme
- Bileşen Tabanlı Geliştirme

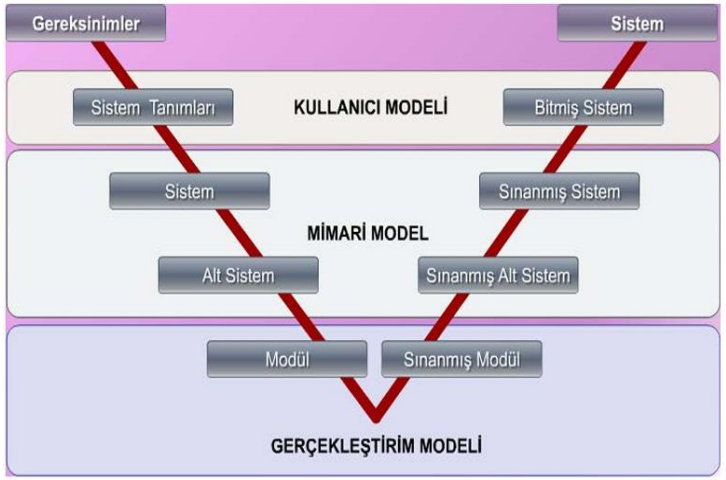
Birleşik Süreç Modeli

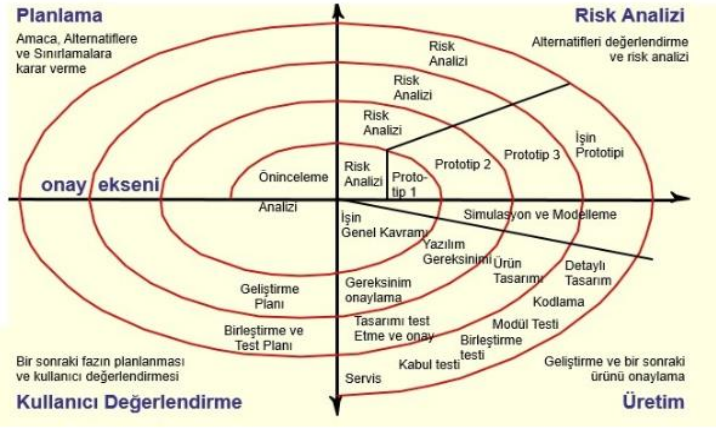
Çevik Süreç Modelleri

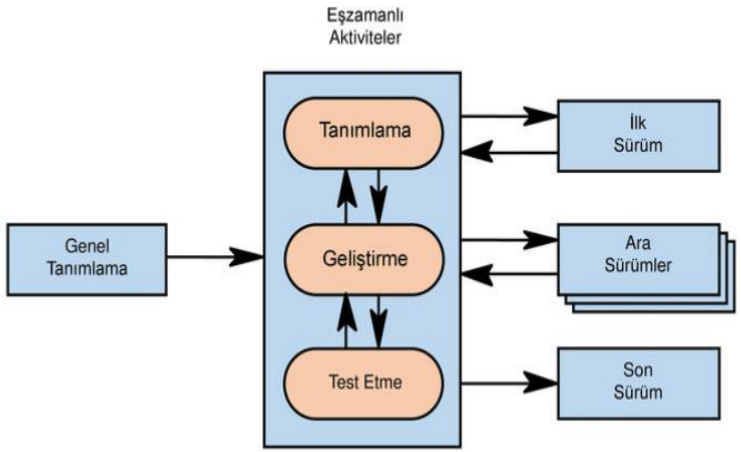
- Uç Programlama
- Scrum
- Özellik Tabanlı Geliştirme
- Çevik Birleşik Süreç
- Kristal Yöntemler
- Dinamik Sistem Geliştirme Yöntemi

Düzenleyici Süreç Modelleri

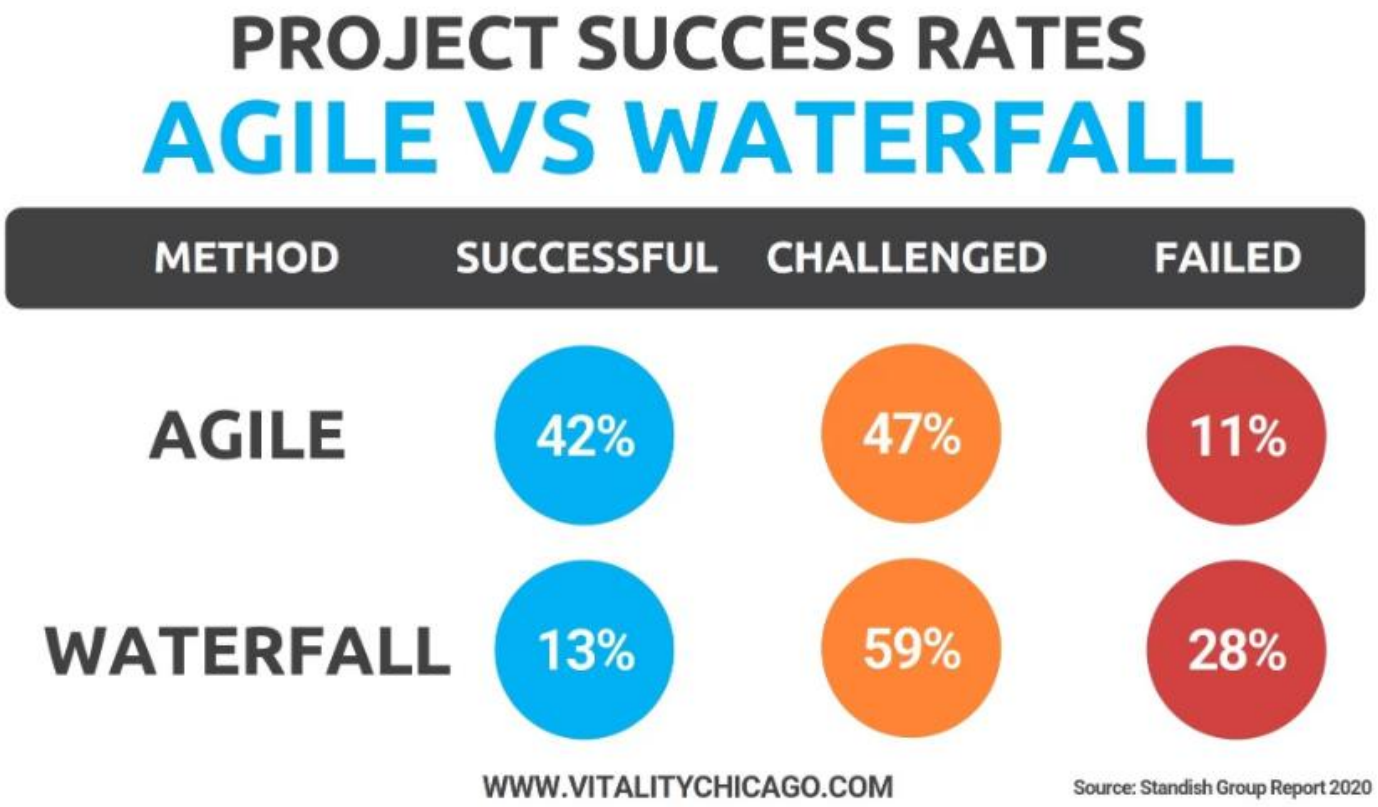








Yazılım Projelerinde Başarı



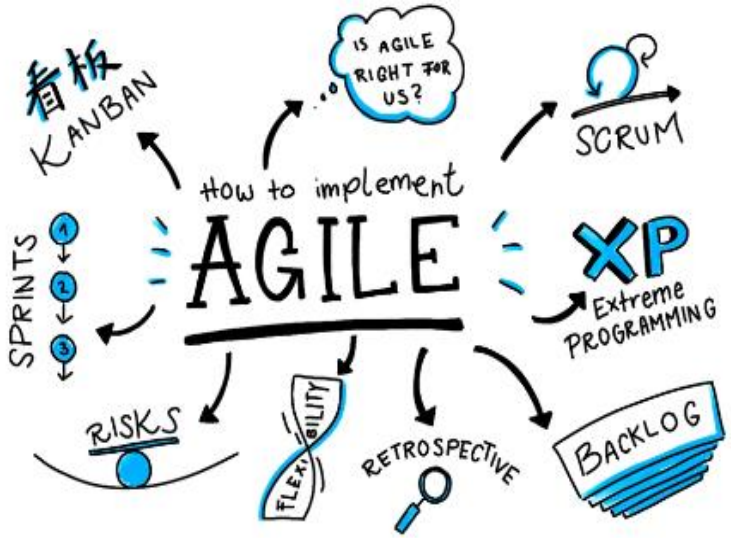
Yazılım Projelerinde Başarısızlığın Sebepleri

1. Müşterinin isteklerini doğru analiz edememek
2. Proje için uygun ekibi kuramamak
3. Gerekli bütçe ve kaynakları ayırmamak
4. Proje yönetim metodları uygulamadan gelişigüzel geliştirmek
5. Proje süresince müşteri ile iletişimden kaçınmak
6. Yanlış teknoloji ya da mimari seçimleri
7. Şirketin yönetimsel sorunları

Çevik (Agile) Yazılım Süreç Modelleri

- Çevik modeller, mevcut geleneksel modellere alternatif olarak, 1990'larda ortaya çıkmaya başlamıştır.
  - 1950'lerdeki üretim alanında verimliliğin artırılması için geliştirilen yalın yaklaşımların yazılım sektöründe bir uzantısı olarak ortaya çıkmıştır.
- Çevik modeller kapsamında yazılım sistemlerini etkili ve verimli bir şekilde modellemeye ve belgelendirmeye yönelik pratiğe dayalı yöntemler yer alır.
- Bu modelleme biçiminin; kapsadığı değerler, prensipler ve pratikler sayesinde geleneksel modelleme metotlarına göre yazılımlara daha esnek ve kullanışlı biçimde uygulanabileceği savunulmaktadır.

Ne zaman tercih edilir?



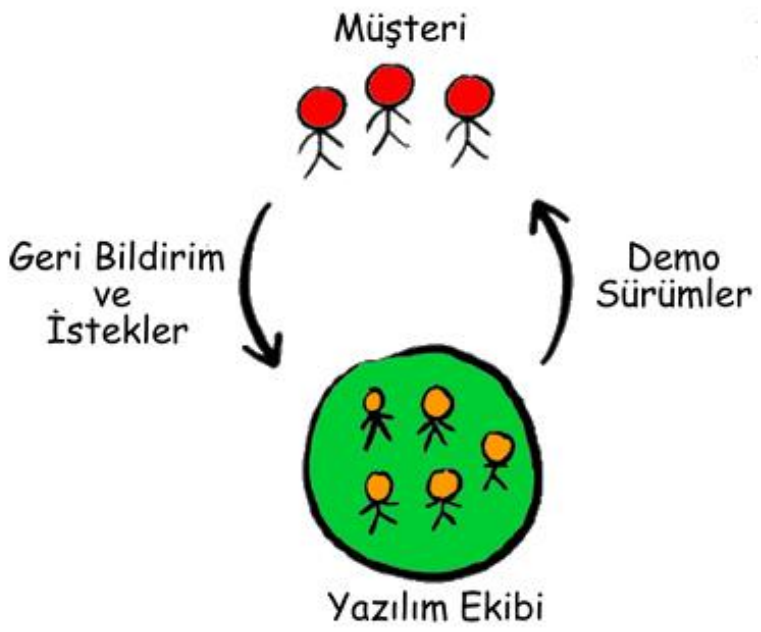
- Projenin yazılım evresinde müşteriden gelebilecek talep değişikliklerinin tahmin edilemez olması
- Projenin parçalarının önce tasarlanıp ardından hemen geliştirilmesinin gerekmesi ve önceden ne yapılacağını, detaylı yol haritasını ve tasarımını tahmin etmenin çok güç olması
- Analiz, tasarım ve test etme süreçlerinin ne kadar zaman alacağının önceden bilinememesi
- Yazılım ekibinin birlikte çalışmak, hiyerarşiye önem vermemek, sağlam iletişim kurmak gibi özelliklere sahip olması

Çevik Yazılım Modelleri

- Bir projenin gerçekleştirilmesinde uyarlanabilir ve tahmin edilebilir model olarak iki tip model kullanılabilir.
- Değişime tepki verebilecek şekilde tasarlanmıştır.
  - Örneğin; projenin gereksinimleri değişirse proje takımı da değişikliğe uyum sağlar ve değişir. Bu takım bir hafta sonra hangi işin yapılacağını söyleyebilir ancak bir ay sonra ne yapacaklarını söyleyemez.
- Projede planlanan tarih ne kadar uzaksa o tarihte yapılacak işler de bir o kadar belirsizdir.
- Tahmin edilebilir olan modellerin takımları projenin tamamlanma süresi içindeki tüm değişikliklerin ve gelişmelerin tarihini önceden bilir ve bu plan çerçevesinde iş yapar.
- Değişiklik olduğunda tüm plan iptal olur ve yeni bir plan yapılır.
- Uyarlanabilir olduğu için net bir şekilde planlanmayan proje müşteri isteklerine göre şekil alır.

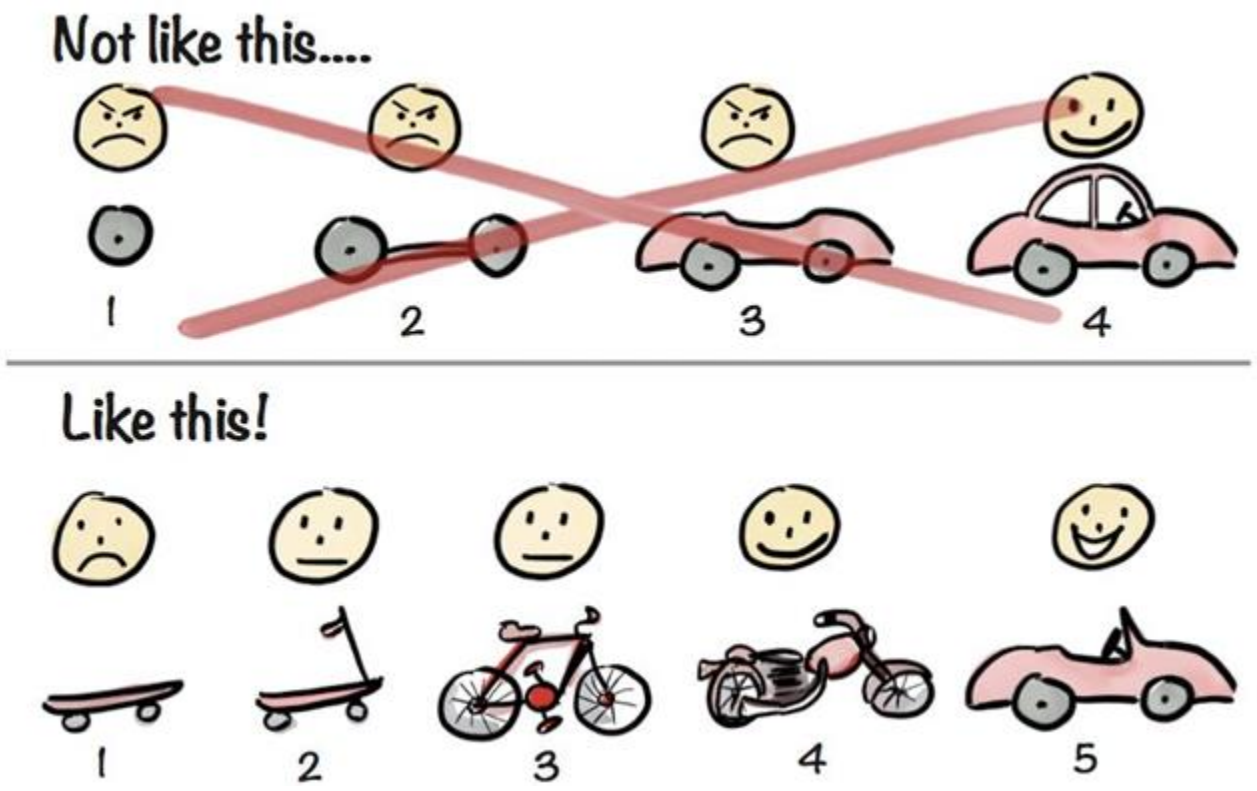
Çevik Yazılım Modelleri

- Çevik yazılım metodu, kısa vadeli planlar ve küçük parçalar halinde yazılımın geliştirilmesini ön görür.
- Yazılımın geliştirilmesindeki geri dönüş (feedback) ve değişikliklere uyum sağlamak son derece önemlidir.
- Her yapılan yineleme yazılımı hedeflenen adıma bir adım daha yakınlaştırır.
- İstenilen sonuca ulaşmak adına birden çok yineleme gereklidir.
- Değişen gereksinimler hakkında hızlı geri bildirim almak için müşterileri geliştirme sürecine dahil eder.

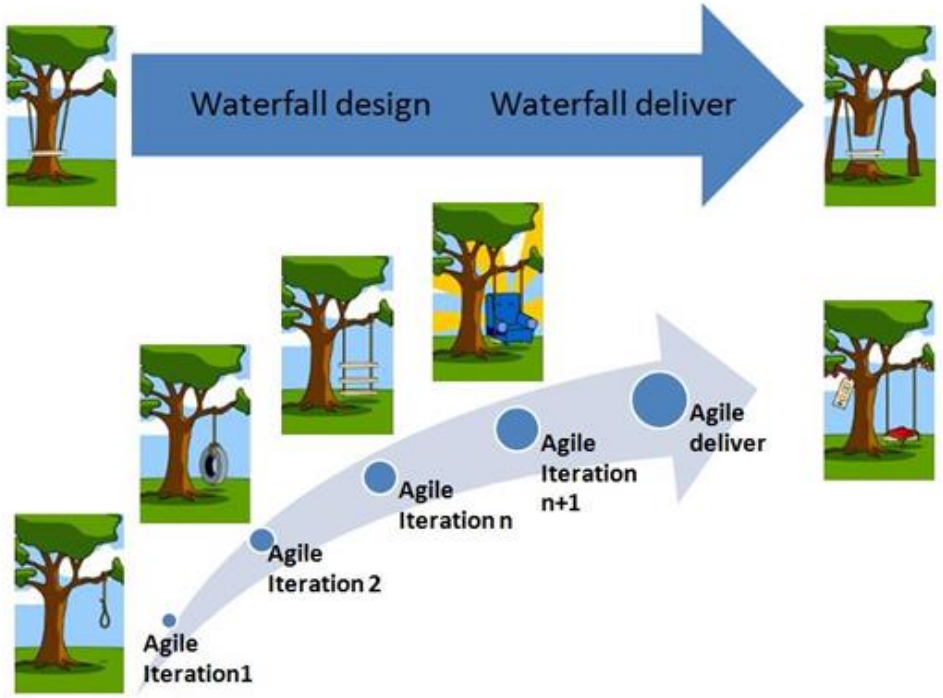


Çevik Yazılım Süreci

Geleneksel Model vs Çevik Model

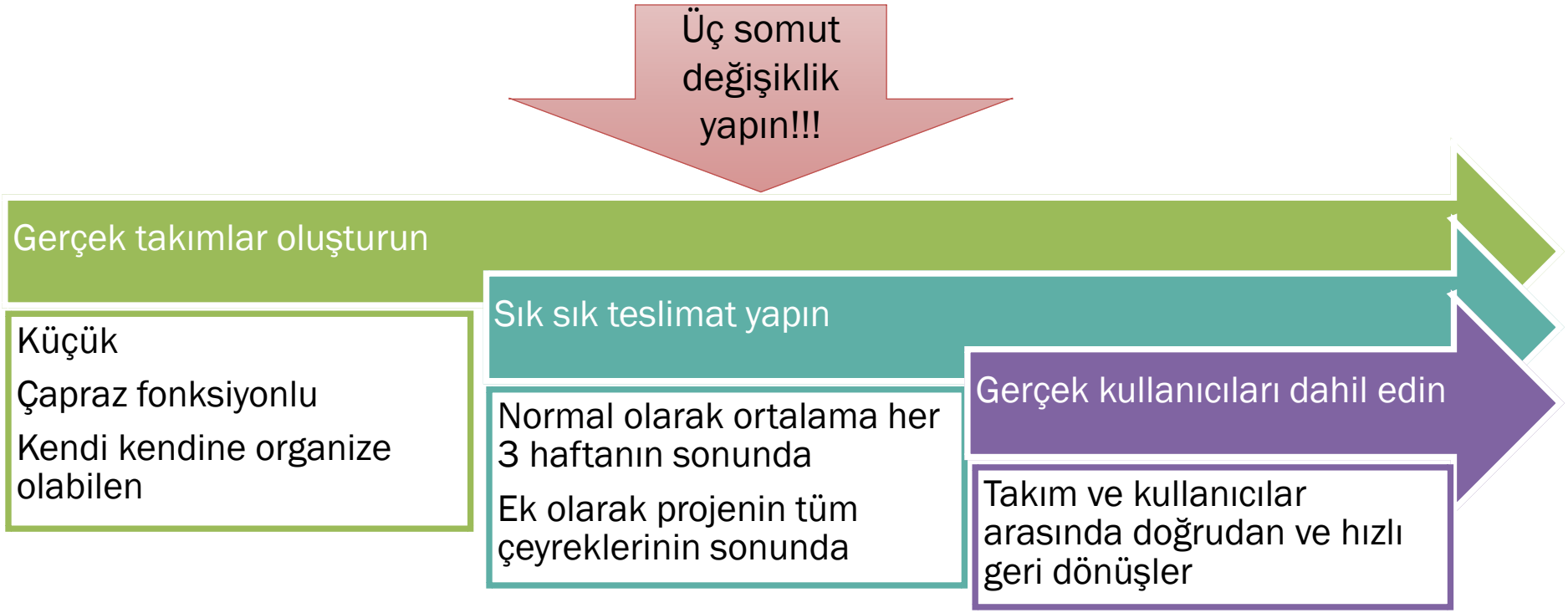


Geleneksel Model vs Çevik Model



Geleneksel Model	Çevik Model
Müşteriler ne istediğini iyi bilir.	Müşteriler ne istediğini keşfeder.
Geliştiriciler neyi, ne şekilde üreteceklerini iyi bilir.	Geliştiriciler neyi, nasıl üreteceklerini keşfeder.
Bu yol boyunca hiçbir şey değişmeyecektir.	Bu yol boyunca birçok değişiklik yapılabilir.

Çevik Yazılım Geliştirme



Çevik Yazılım Manifestosu

Bireyler ve aralarındaki etkileşim	>	Kullanılan araçlar ve süreçler
Çalışan yazılım ve çözümler	>	Kapsamlı dökümantasyon
Müşteriyle iş birliği	>	Sözleşmelere bağlı kalmak
Değişime cevap verebilmek	>	Herhangi bir planı izlemek

daha önemli

az önemli

Çevik Yazılım Prensipleri

1.Maksimum müşteri memnuniyeti

2.Adaptasyon – Değişimi hoş karşıla

3.Sık teslimat yap

4.Sürekli müşteri ile birlikte çalış

5.Güven ve destek ortamı ile insanları motive et

6.Yüz yüze iletişim kur

7.Çalışan yazılım

8.Sürdürebilirliği önemse

9.Tasarımını sürekli iyileştir

10.Sade ol

11.Kendi kendine organize olan ekipler kur

12.Sürekli iyileştir

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

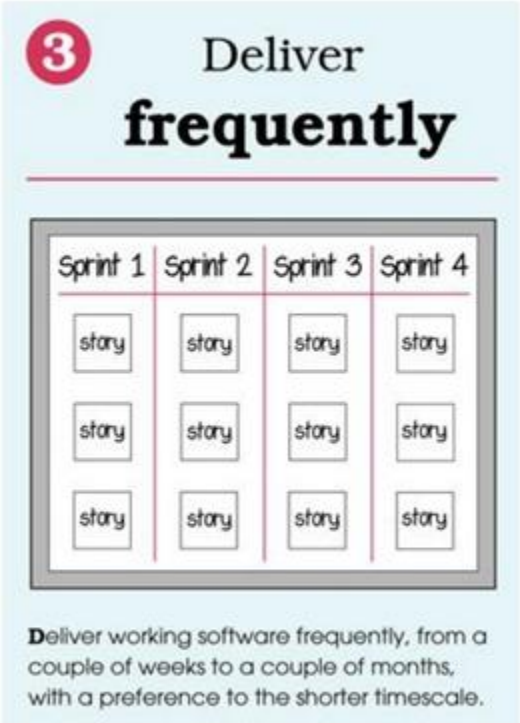
En önemli önceliğimiz değer ifade eden çıktının erken ve devamlı teslimini sağlayarak müşterileri memnun etmektir.

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

Değişen gereksinimler geliştirme sürecinin son aşamalarında bile kabul edilmelidir. Çevik süreçler, değişimi müşterinin rekabet avantajı için kullanır.

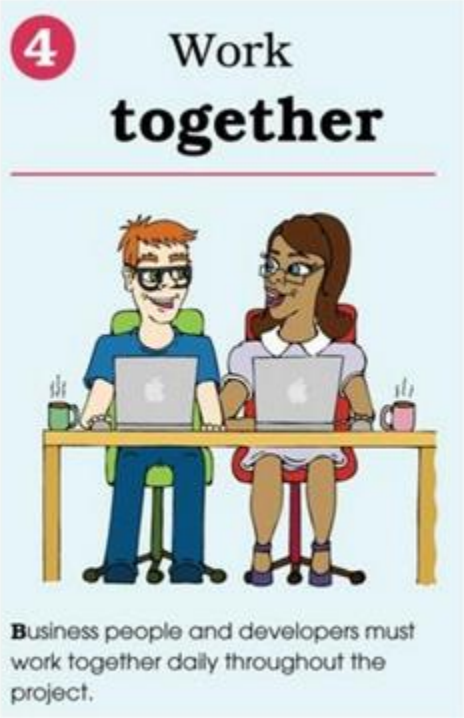
Çevik Yazılım Prensipleri



http://agilemanifesto.org/principles.html

Çalışan yazılım, tercihen kısa zaman aralıkları belirlenerek birkaç haftada ya da birkaç ayda bir düzenli olarak müşteriye sunulmalıdır.

Çevik Yazılım Prensipleri



http://agilemanifesto.org/principles.html

İş süreçlerinin sahipleri ve geliştirme ekipleri proje boyunca her gün birlikte çalışmalıdırlar.

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

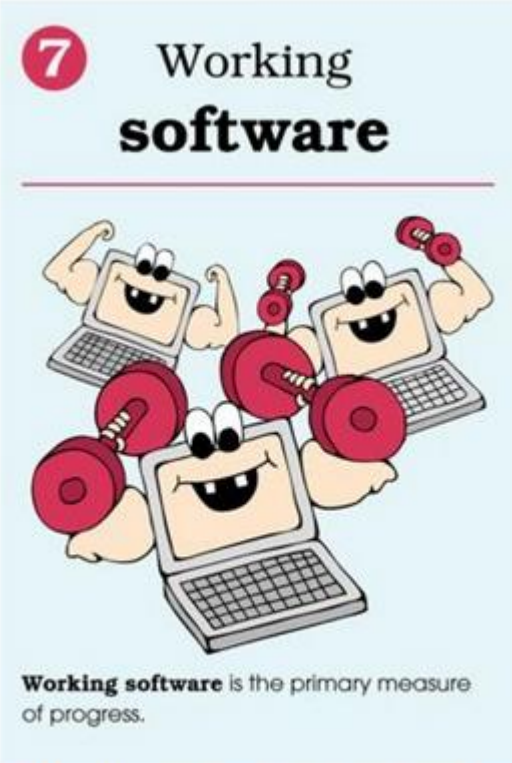
Projelerin temelinde motive olmuş bireyler yer almalıdır. Onlara ihtiyaçları olan ortam ve destek sağlanmalı, işi başaracakları konusunda güven duyulmalıdır.

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

Bir yazılım takımında bilgi alışverişinin en verimli ve etkin yöntemi yüz yüze iletişimdir.

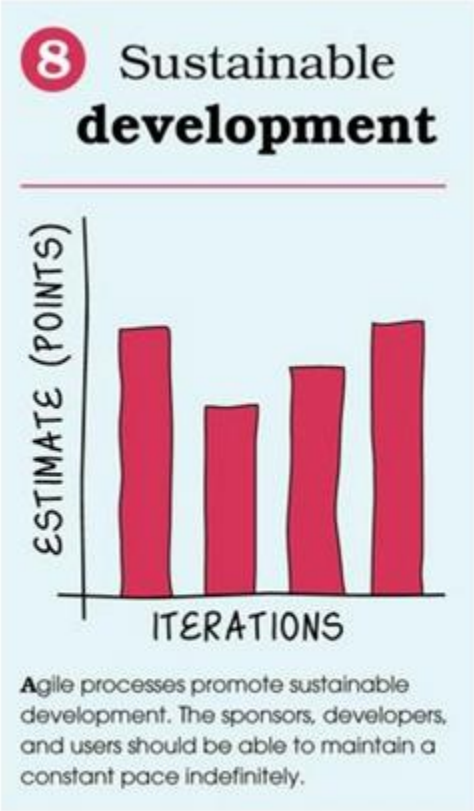
Çevik Yazılım Prensipleri



http://agilemanifesto.org/principles.html

Çalışan yazılım, ilerlemenin birincil ölçüsüdür. Neredeyse tamamlanmış olması bittiği anlamına gelmez.

Çevik Yazılım Prensipleri



http://agilemanifesto.org/principles.html

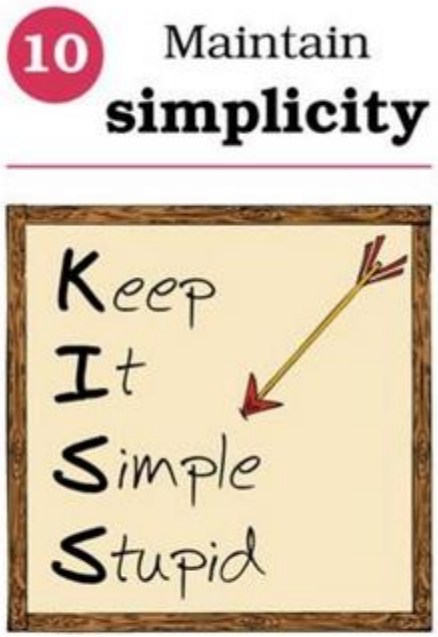
Çevik süreçler sürdürülebilir geliştirmeyi teşvik etmektedir. Sponsorlar, yazılımcılar ve kullanıcılar sabit tempoyu sürekli devam ettirebilmelidir.

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

Teknik mükemmeliyet ve iyi tasarım konusundaki sürekli özen çevikliği artırır.

Çevik Yazılım Prensipleri

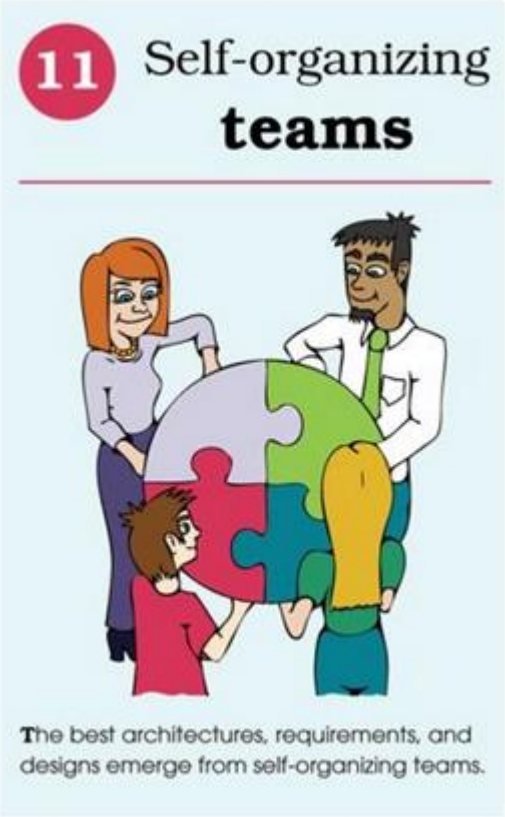


The art of maximizing the amount of work not done - is essential.

http://agilemanifesto.org/principles.html

Sadelik, yapılmasına gerek olmayan işlerin mümkün olduğunca arttırılması sanatı, olmazsa olmazlardandır.

Çevik Yazılım Prensipleri



http://agilemanifesto.org/principles.html

En iyi mimariler, gereksinimler ve tasarımlar kendi kendine organize olabilen takımlardan ortaya çıkar.

Çevik Yazılım Prensipleri

http://agilemanifesto.org/principles.html

Takım, düzenli aralıklarla nasıl daha etkili ve verimli olabileceğinin üzerinde düşünür ve davranışlarını buna göre ayarlar ve düzenler.