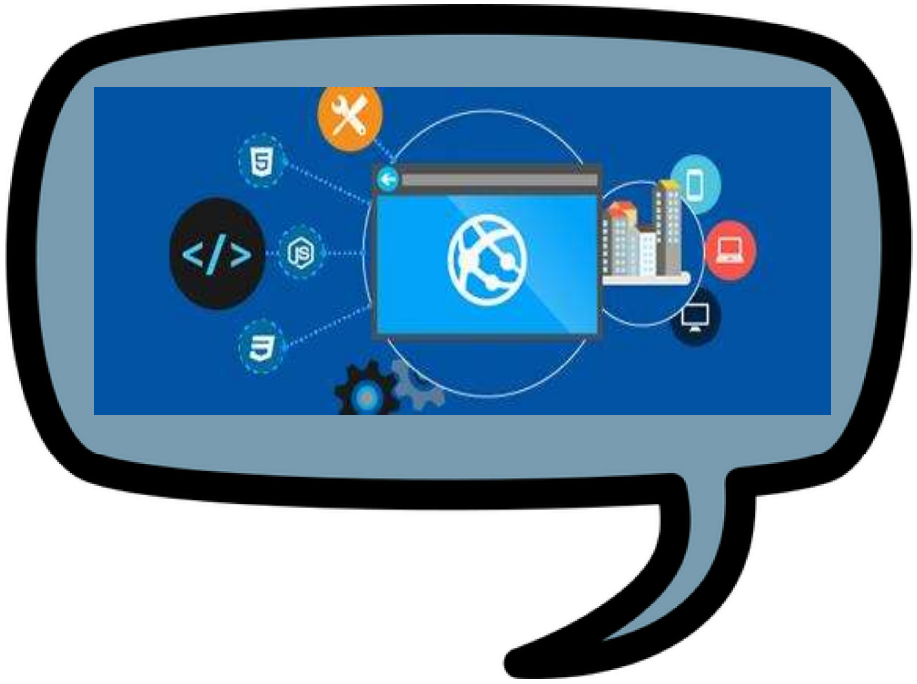
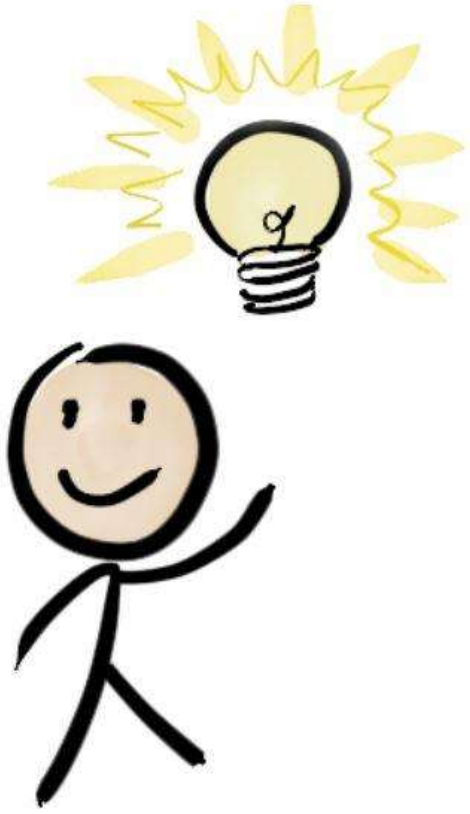
İçindekiler

1.Günümüzde Yazılım Projelerinin Durumu

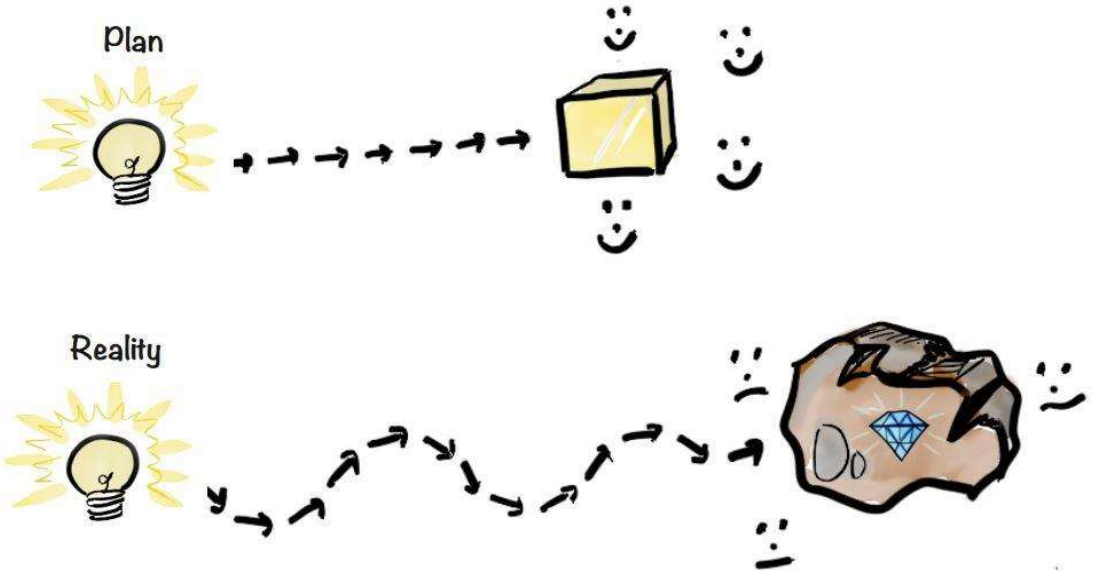
- Birçok proje harika bir fikir ile başlar!





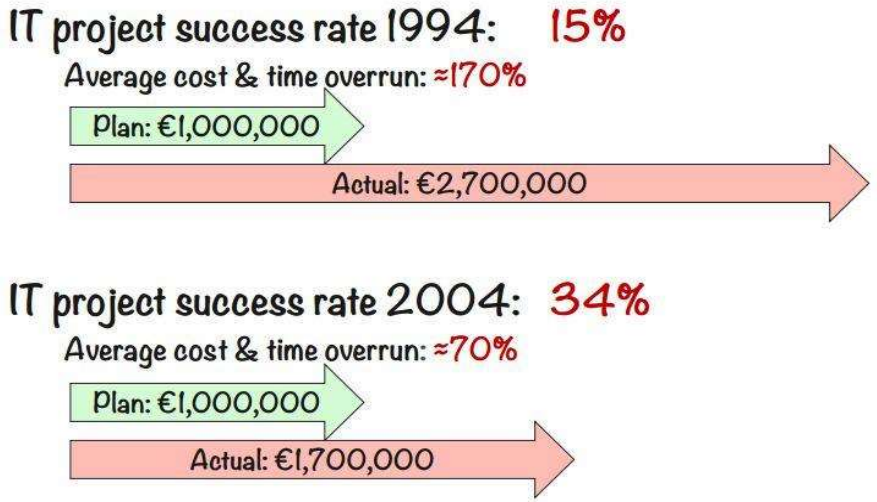
1.Günümüzde Yazılım Projelerinin Durumu

- Bu projelerin büyük bir kısmının başarısız olması muhtemeldir!



1.Günümüzde Yazılım Projelerinin Durumu

- Birçok Bilgi Teknolojisi projesi başarısız olmuş veya gecikmiştir. The Standish Group, 10 yıl içerisinde 40.000'den fazla proje üzerinde çalışmıştır.



1.Günümüzde Yazılım Projelerinin Durumu

Ülkemizde durum nasıl?

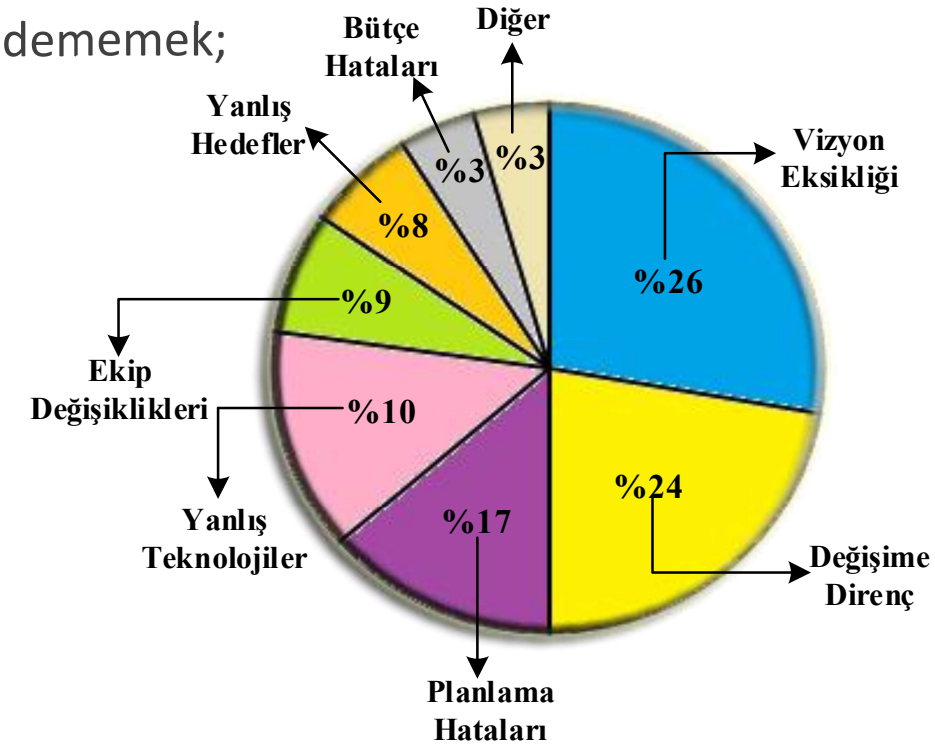
Durum	Oran
Tam başarılı	%4-5
Kısmen başarılı	%45-50
Çöpe gidenler	%50

Agile Turkey

Örneğin, ülkemizde geliştirilen projelerin başarıya ulaşmasına katkı sağlamak amacıyla, Araştırma Destek Programları Başkanlığı (ARDEB) tarafından desteklenen projelerin çıktı, sonuç ve etkilerini nicelik ve nitelik olarak artırmak amacıyla yüksek başarı ile sonuçlanan projelerin yürütücü ve araştırmacılarını ödüllendirmek için TÜBİTAK tarafından belirlenen ölçütler ve değerlendirme yöntemine göre hesaplanarak, proje ekibine (yürütücü ve araştırmacılara) TÜBİTAK Proje Performans Ödülü (PPÖ), denilen bir teşvik ödülü verilmektedir.

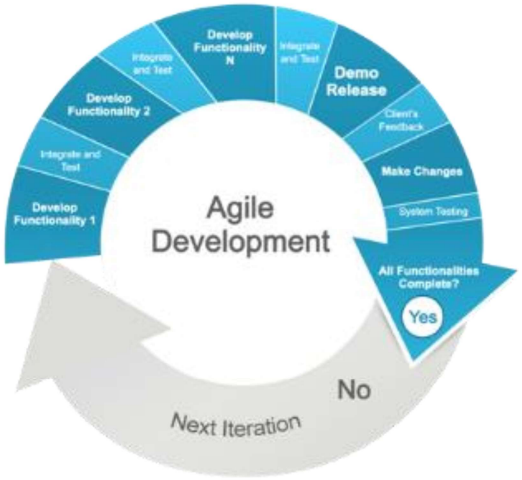
1.Günümüzde Yazılım Projelerinin Durumu

- Başarısızlığın ana sebepleri:
  - Müşterinin isteklerini doğru analiz edememek;
  - Proje için uygun ekibi kuramamak;
  - Yanlış teknoloji ve mimari seçimleri;
  - Geleneksel yöntemlerin eksiklikleri;
  - Müşteriyle iletişimden kaçınmak vs.



1.Günümüzde Yazılım Projelerinin Durumu

Peki ne yapmalıyız?



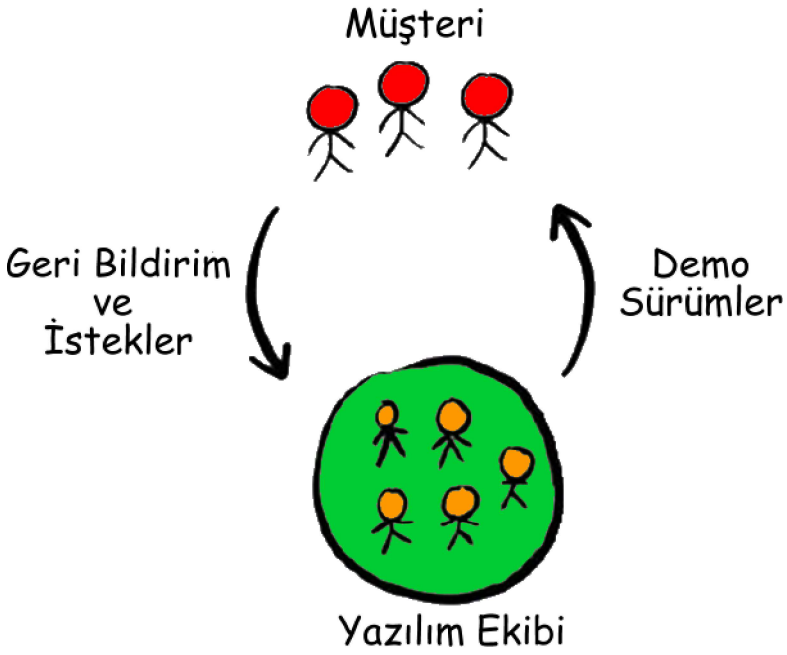
2.Çevik Yazılım Yöntemi

- ***Tekrarlamalı*** ve ***artımsal*** bir ürün geliştirme yöntemidir.
  - **Bireyler ve etkileşimi**, süreç ve araca tercih eder.
  - **Çalışan bir yazılımı**, detaylı ürün belgelendirmeye tercih eder.
  - **Müşteri ile işbirliğini**, sözleşmedeki kesin kurallara tercih eder.
  - **Değişikliklere uyum sağlayabilmeyi**, belirli bir plana tercih eder.

2.Çevik Yazılım Yöntemi

- Hızlı, devamlı ve kullanışlı yazılım üreterek müşteri memnuniyeti sağlamayı amaçlar.
- Geliştiriciler ile iş adamları arasında günlük ve yakın işbirliği bulunmalıdır.
- Çalışan yazılım gelişimin en önemli ölçüsüdür.
- Taleplerdeki geç değişikliklerin de memnuniyetle karşılanır.
- Yüz yüze görüşme iletişimin en güzel yoludur.
- Kendi kendini organize eden takım yapısı gereklidir.
- Basitlik önemlidir.

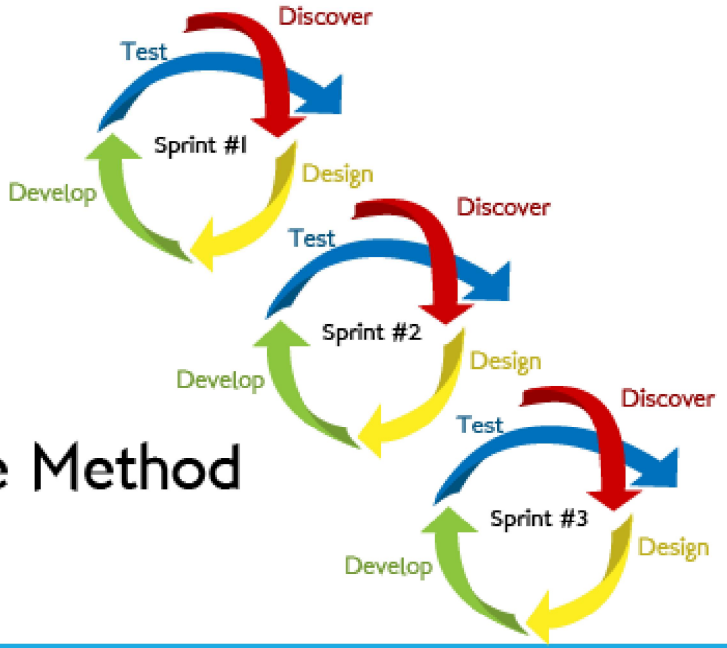
2.Çevik Yazılım Yöntemi



- 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.

2.Çevik Yazılım Yöntemi

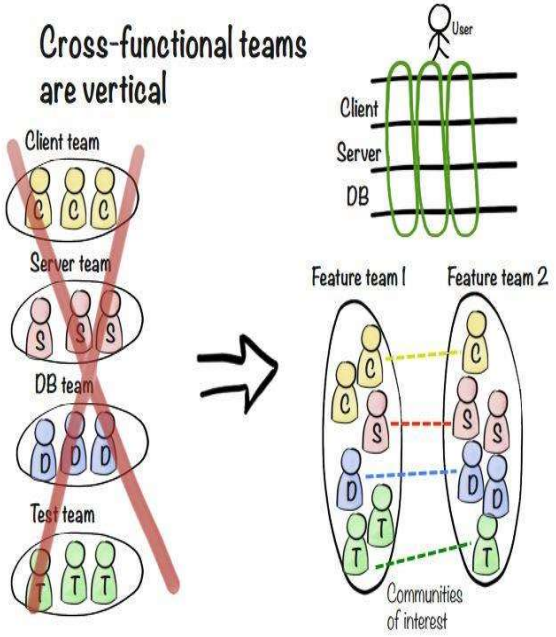
- Örnek:



2.Çevik Yazılım Yöntemi

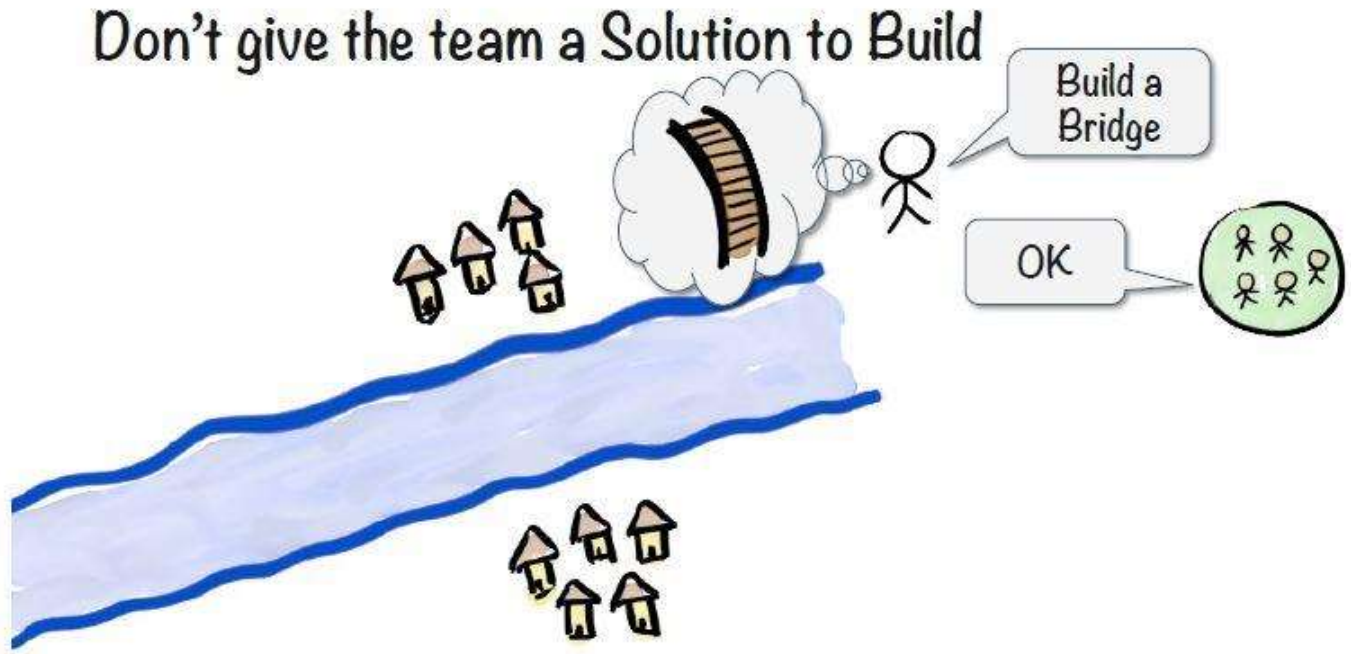
- Temel prensipler:
- Müşteriyi memnun etmek
- Değişen ihtiyaçları karşılamak
- Sık aralıklarla ürün teslimi yapmak
- Yüz yüze iletişime önem vermek
- Sürdürülebilir gelişmeyi desteklemek
- Teknik mükemmeliyete, iyi dizayna ve sadeliğe odaklanmak
- **Kendi kendine organize olan takımlar kurmak ?**

2.1.Çevik Model Takımları



- Biraraya gelmiş,
- Kendi kendilerine organize olan,
- Çapraz fonksiyonlu,
- İşine odaklanmış,
- Hedefleri net olan,
- Teslim edilebilecek düzeyde ürün ortaya koyabilen
- Küçük(3-7 kişilik) gruplar.

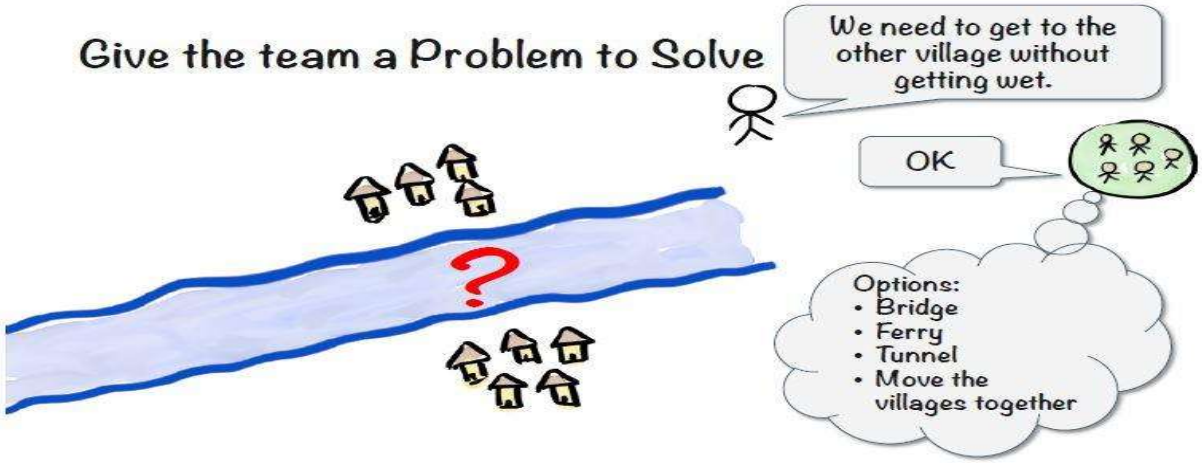
2.1.Çevik Model Takımları



Takımlara çözümü söylemeyin!

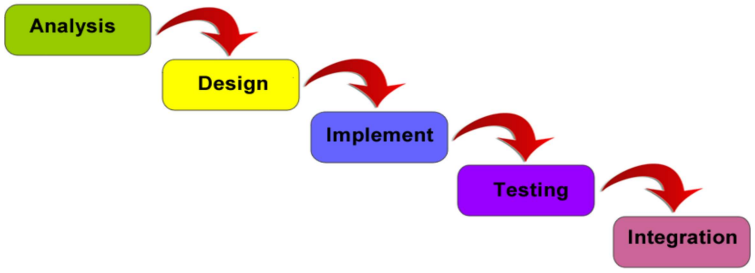
2.1.Çevik Model Takımları

Sorunu söyleyin, onlar çözümü üretsin!



3. Geleneksel Model vs. Agile

- Çağlayan modeli 2008 yılında dahi geçerliliğini koruyan bir modeldir ve çevik modellemeden farklılık gösterir. Bu model yazılım projesini baştan sona planlar. Gelişim, sunulabilir işler açısından ölçülür: talep açıklamaları, tasarım dokümanları, test planları, kod incelemeleri vb. Bu durum belli aralıklara bölünmeye uygun değildir ve ilerideki değişikliklere uyum gösterilemez.

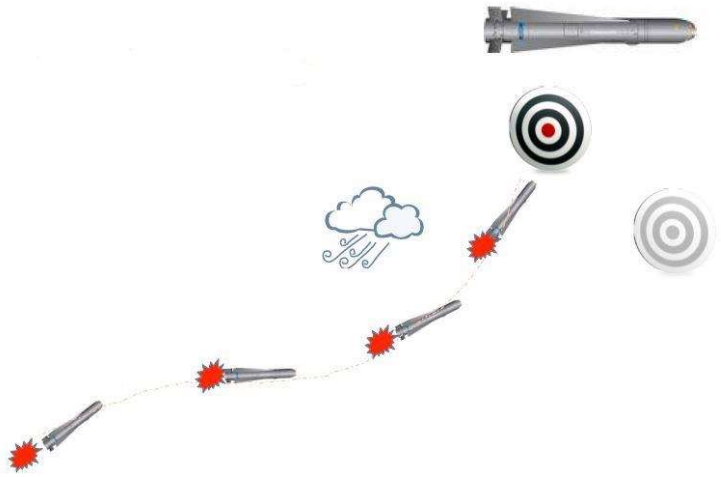


Waterfall Model

3. Geleneksel Model vs. Agile

- Geleneksel Yöntemler
- Müşteriler ne istediğini iyi bilir.
- Geliştiriciler neyi, ne şekilde üreteceklerini iyi bilir.
- Bu yol boyunca hiç birşey değişmeyecektir.

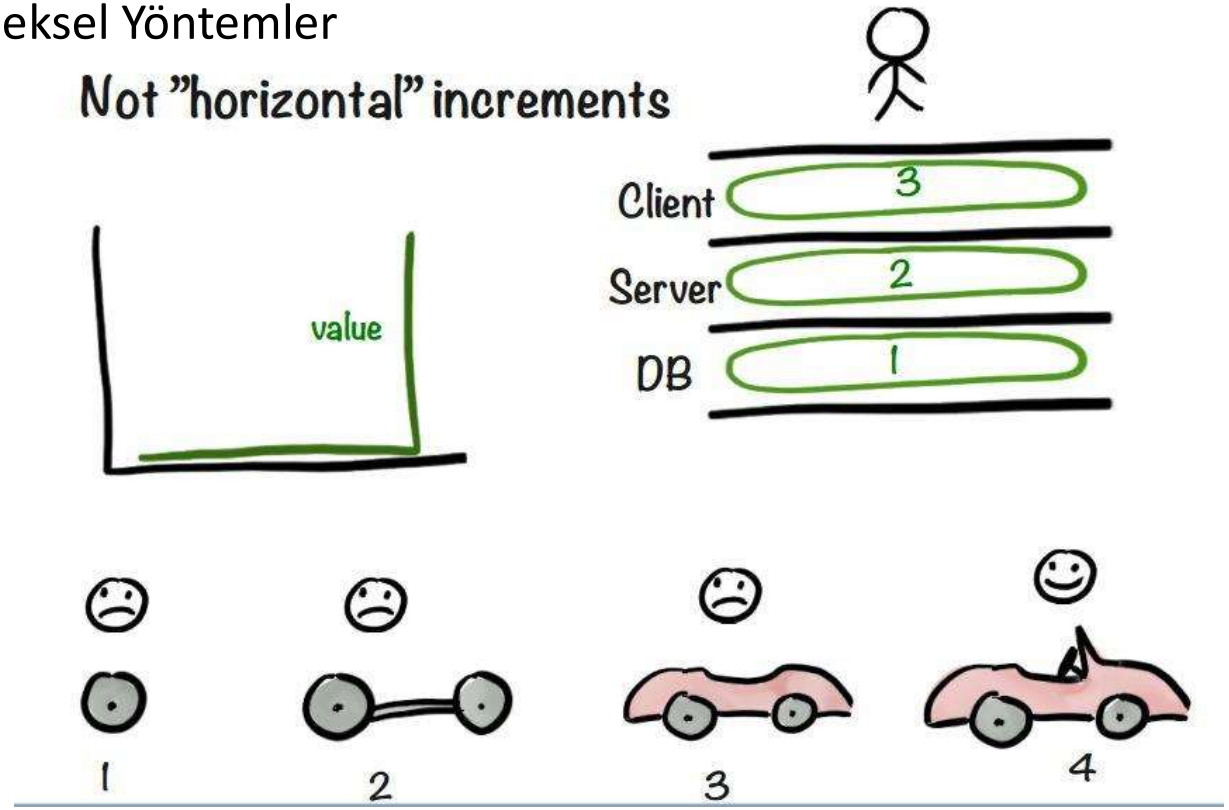
3. Geleneksel Model vs. Agile



- Çevik Yöntemler
- Müşteriler ne istediğini keşfeder.
- Geliştiriciler neyi nasıl üreteceğini keşfeder.
- Bu yol boyunca bir çok değişiklik yapılabilir.

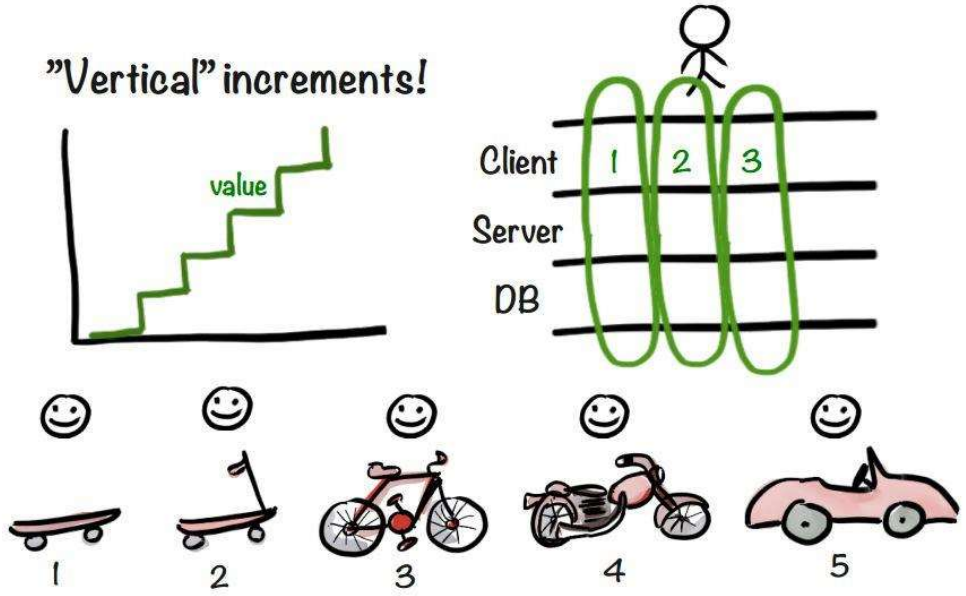
3. Geleneksel Model vs. Agile

Geleneksel Yöntemler

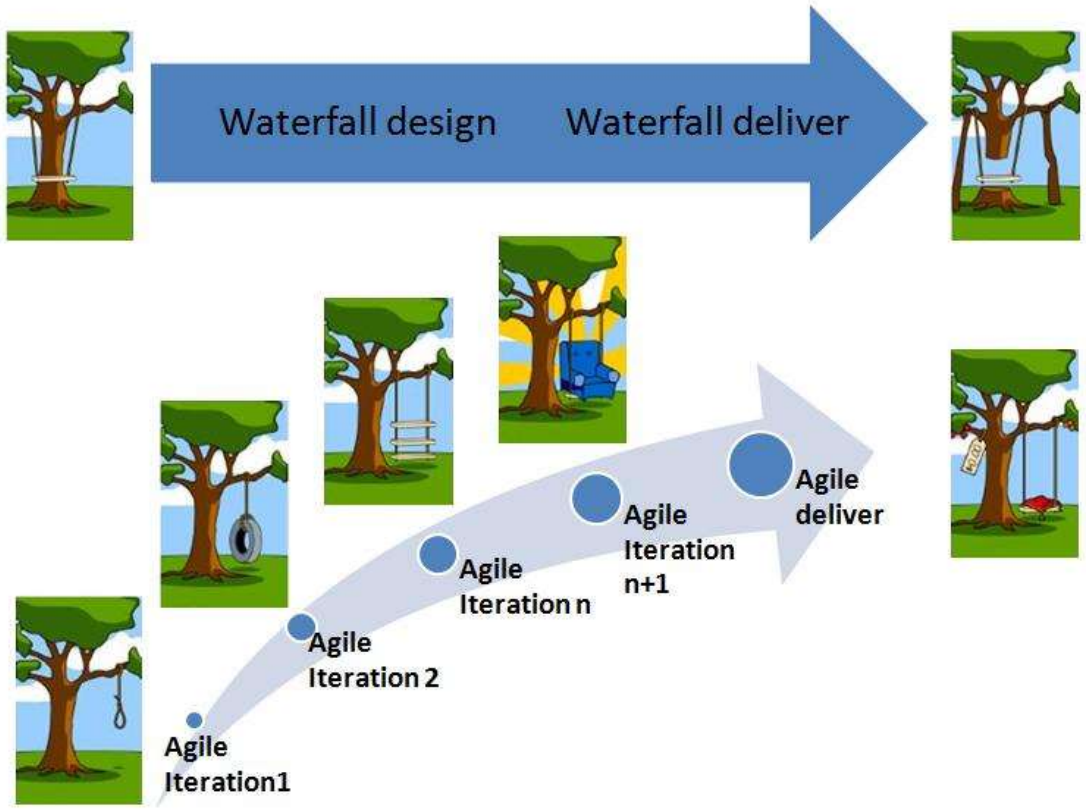


3. Geleneksel Model vs. Agile

Çevik Yöntemler

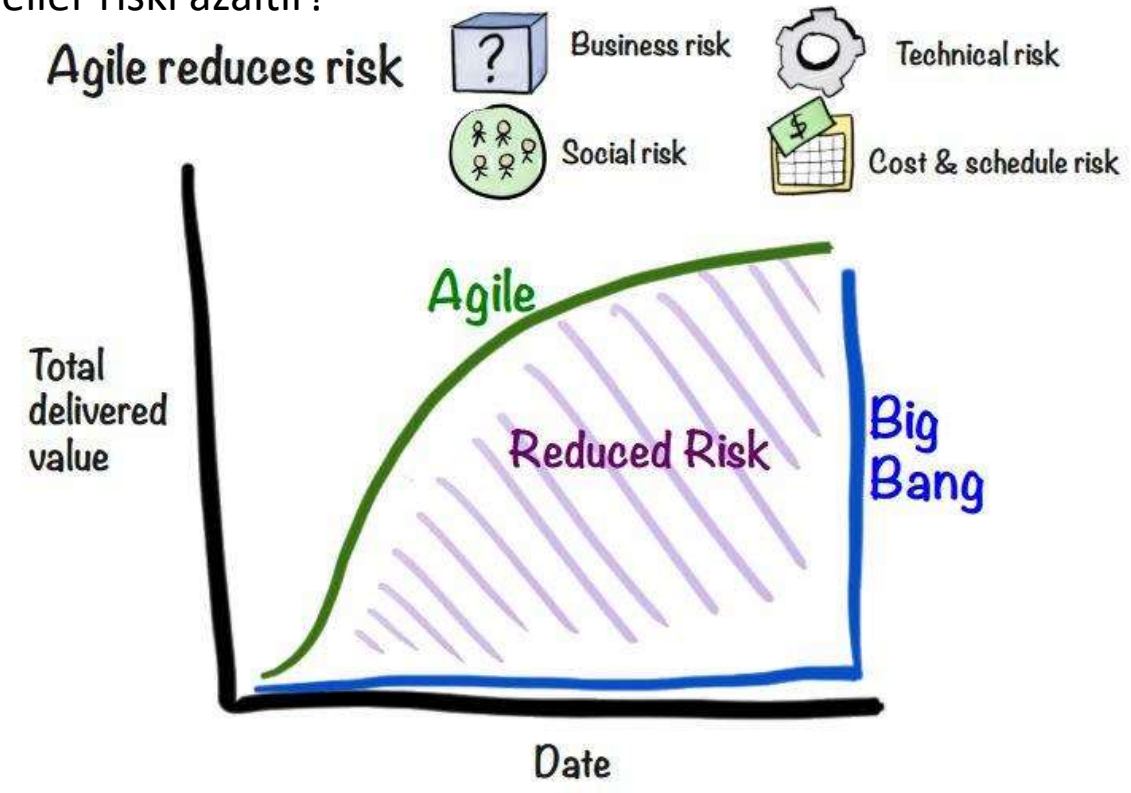


3. Geleneksel Model vs. Agile

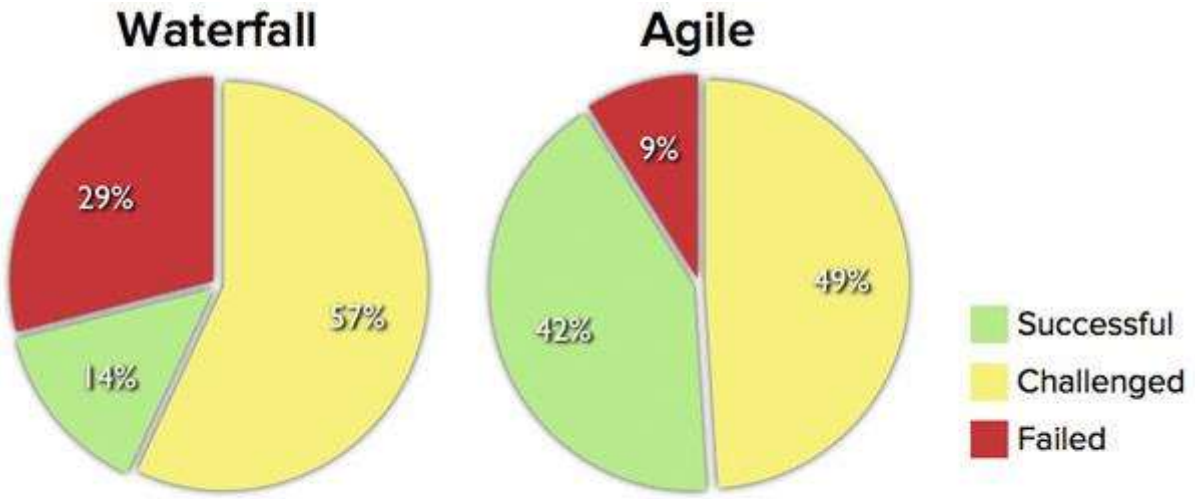


3. Geleneksel Model vs. Agile

Çevik modeller riski azaltır!



3. Geleneksel Model vs. Agile

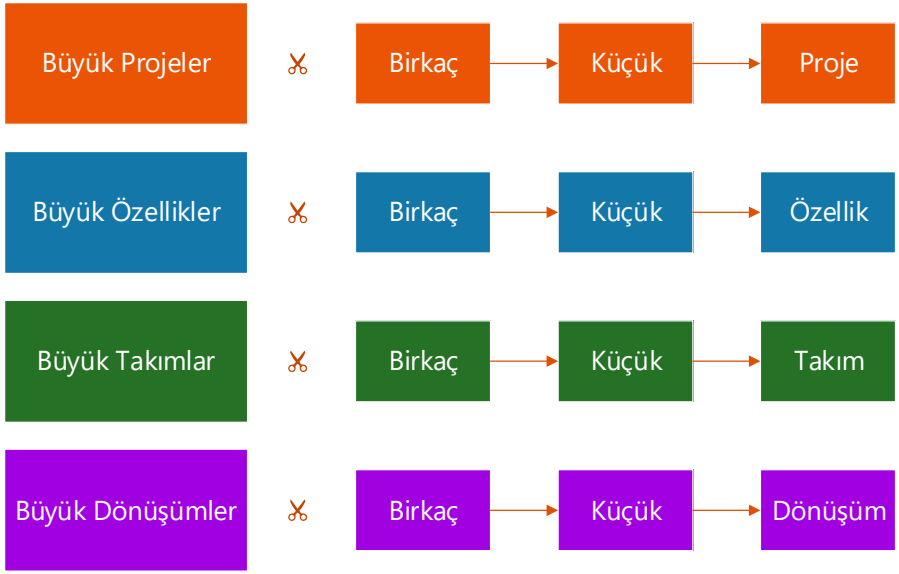


Source: The CHAOS Manifesto, The Standish Group, 2012.

3. Geleneksel Model vs. Agile

Ölçüm	Çevik Modelleme	Çağlayan Modeli
Planlama ölçeği	Kısa dönemlik	Uzun dönemlik
Müşteri ile geliştirici arasındaki mesafe	Kısa	Uzun
Özelleştirme ve uygulama arasındaki zaman	Kısa	Uzun
Sorunları keşfetmek için zaman	Kısa	Uzun
Proje tamamlanma riski	Düşük	Yüksek
Değişikliklere uyum yeteneği	Yüksek	Düşük

4. Değerlendirme



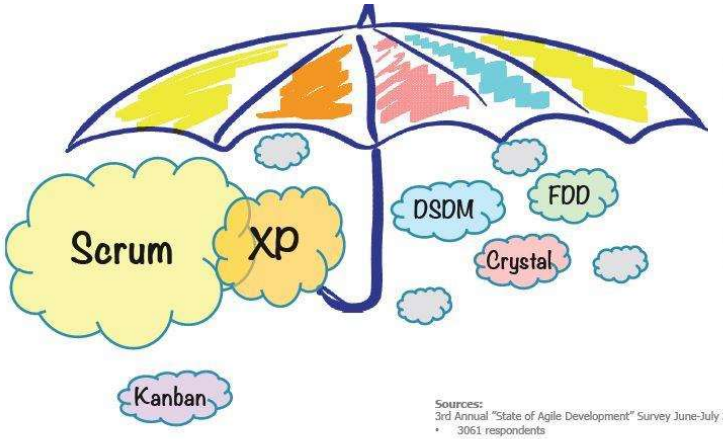
Sonuç

- Büyüklük kötüdür, onu parçalara ayırın. Böylece daha başarılı projeler geliştirebilirsiniz.

4. Değerlendirme

- 3 somut değişiklik yapın!
- 1. Gerçek takımlar oluşturun
  - Küçük, çapraz fonksiyonlu, kendi kendine organize olabilen
- 2. Sık sık teslimat yapın
  - Normal olarak ortalama her 3 haftanın sonunda
  - Ek olarak projenin tüm çeyreklerinin sonunda
- 3. Gerçek kullanıcıları dahil edin.
  - Takım ve kullanıcılar arasında doğrudan ve hızlı geri dönüşler

5. Çevik Yazılım Şemsiyesi



Sources:
3rd Annual "State of Agile Development" Survey June-July 2008
- 3061 respondents
- 80 countries

FDD: Feature-Driven Development

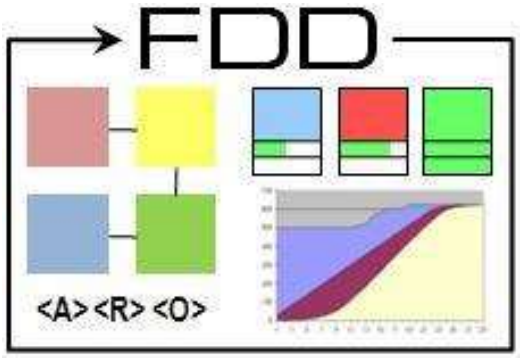
RUP : Rational Unified Process

DSDM : Dynamic System Development Method

FDD TANIMI

- **FDD, Avustralyalı Jeff De Luca tarafından geliştirilmiş ve Singapur projesinde beraber çalıştıkları Peter Coad tarafından modifiye edilmiştir. Singapur projesi 50 kişi ile 15 ayda tamamlanan bir proje olmuştur. Daha sonra FDD'nin uygulandığı başka bir proje ise 250 kişi ile 18 ayda tamamlanmıştır. FDD değişik boyutlara büyüyebilen, tekrarlanabilir bir süreçtir. Aşağıdaki noktalara odaklanır.**

1. **Sistemi hazırlamak için gereken sistem büyüyebilir olmalıdır. Büyük projeler için de kullanılabilir olmalıdır.**
2. **Basit iyi tanımlanmış sistem iyi çalışır.**
3. **Süreç adımları basit olmalıdır.**
4. **İyi süreç arka plana saklanır ve insanlar sonuçlara odaklanabilir.**
5. **Kısa, iteratif, özellik yaklaşımlı yaşam döngüleri en iyi sonucu verir**



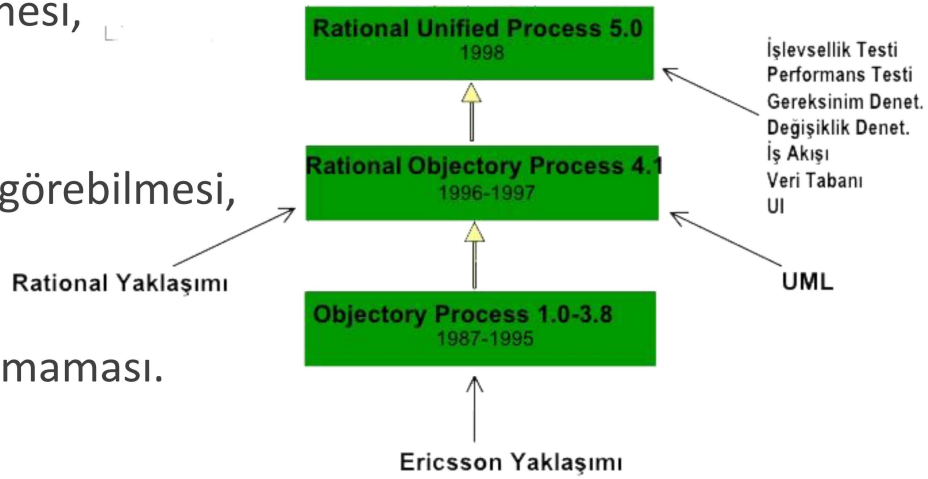
RUP ("Rational Unified Process")

- 2003 yılından beri IBM'in bir bölümü tarafından oluşturulan bir iteratif yazılım geliştirme süreci çerçevesidir. Başarısız bir yazılımdaki sorunların aşılıp başarılı yazılım oluşturmak için gerekli adımları saptayarak oluşturulmuş bir süreçtir.

- Başarısız bir yazılımdaki özelliklerini yazımızın devamında okuyabilirsiniz. RUP şirketlere yazılım geliştirme aşamasında bir yön sağlayar. RUP use-case ve nesne teknolojileri tabanlı; tekrarlanan (iterative) yazılım geliştirme ve iş modelleme yöntemidir. RUP'un verebileceği özellikler şunlardır;

1. Müşteriyi ve yazılımcıyı organize edebilmesi,
2. Standart tanımlı adımları olması,
3. Oluşacak yazılımdaki sık değişiklikleri öngörebilmesi,
4. Basit olması,
5. Proje yönetim aktivitelerinin çok fazla olmaması.

RUP'un Gelişimi



Uç Programlama (Extreme Programming XP) Nedir?

➢ **Uç Programlama (XP)**, yazılım geliştirme süreci boyunca son derece *kaliteli olmak koşuluyla* **çalıştırılabilir kod** üretmeye odaklanmış bir yazılım geliştirme metodolojisidir. Yazılım geliştirme sürecinin en temel, en önemli ve final çıktısı ya da ürünü çalıştırılabilir kod olduğundan, XP metodolojisi sürecin en başından itibaren çalıştırılabilir kodu sürecin merkezinde tutmaktadır. İşte bu yüzden bu metodolojinin adı XP'dir.

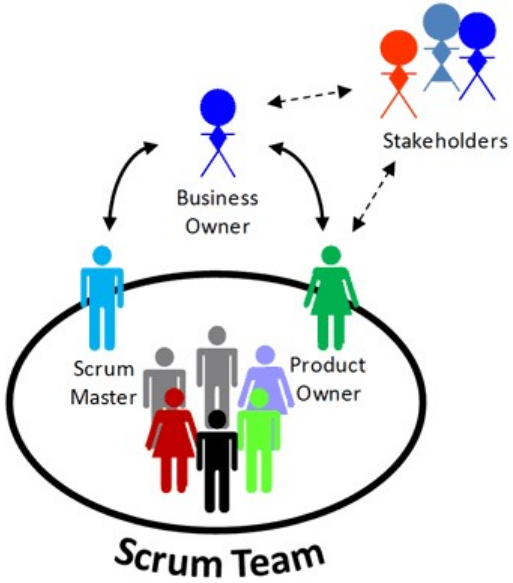
Uç Programlama'nın özünde aşağıdaki uygulamalar yer alır:

- Planlama
- Sık ve küçük sürümler
- Basit tasarım:
- Önce test
- Refactor etme
- Haftada 40 saat çalışma
- Müşteriyle yakın iletişim
- Kodlama standartları

6. Scrum Modeli

Scrum Süreci

6. Scrum Modeli



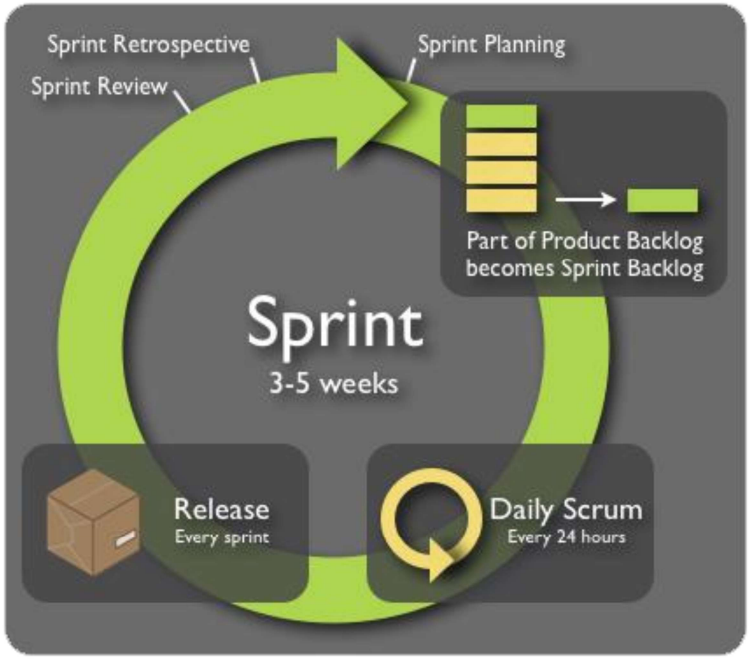
- Scrum Takımı: Ürün Sahibi, Geliştirme Ekibi ve Scrum Master'dan oluşur. Takım kendi kendini örgütler. Böylece kendi içerisinde uyum içinde olan takımlar daha başarılı sonuçlar alırlar. Scrum takım modeli esneklik, yaratıcılık ve verimliliği optimize etmek için tasarlanmıştır.

6. Scrum Modeli

Backlog:

- Müşteriden ve son kullanıcıdan gelen gereksinimleri içerir.
- "Ne yapacağız" sorusunun yanıtını içerir.
- Herkese açık ve herkes tarafından müdahale edilebilir.
- Risk, iş değeri, zaman gibi kavramlara göre ürün sahibi tarafından sıralandırılır.
- User Story'lerden oluşur.

6. Scrum Modeli



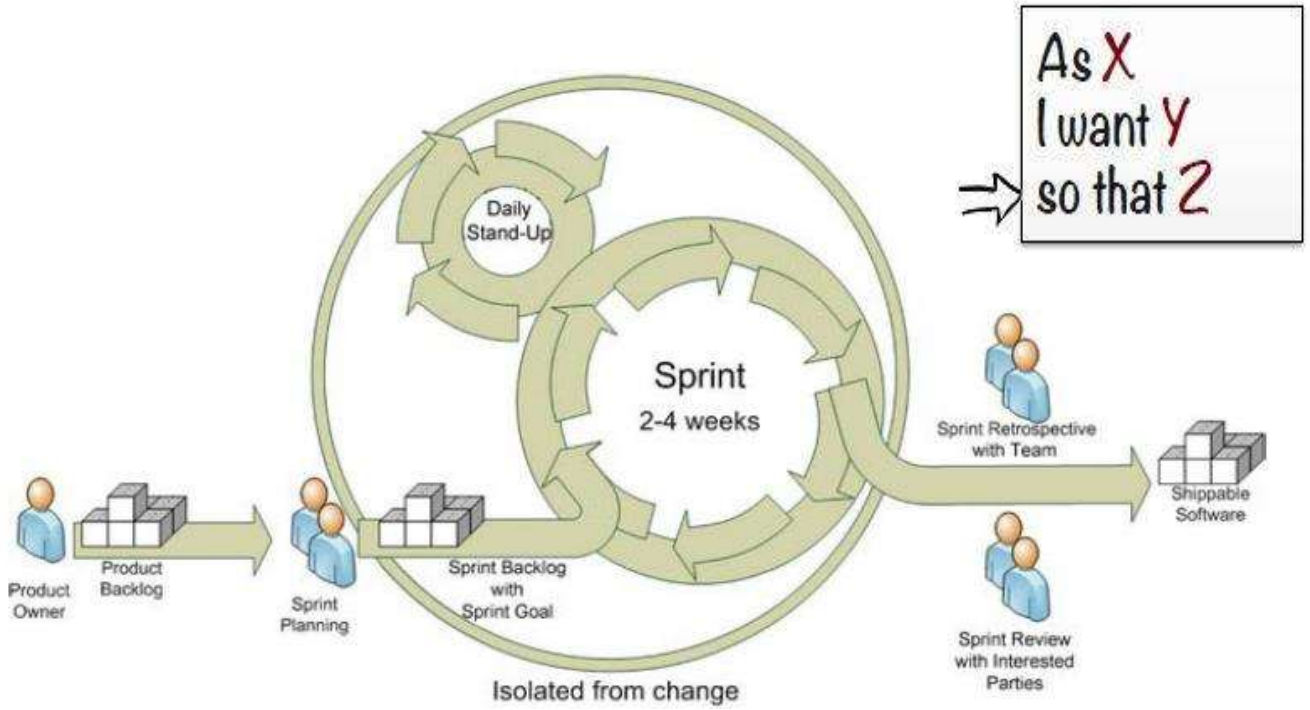
- **Sprint**
- Belirli bir süreye sahiptir.
- Sonunda ortada değeri olan bir çıktı olmalıdır.
- Toplantılarla içerik belirlenir.
- Sprint süresi boyunca her gün toplantılar yapılır.

6. Scrum Modeli

Sprint Gösterimi

6. Scrum Modeli

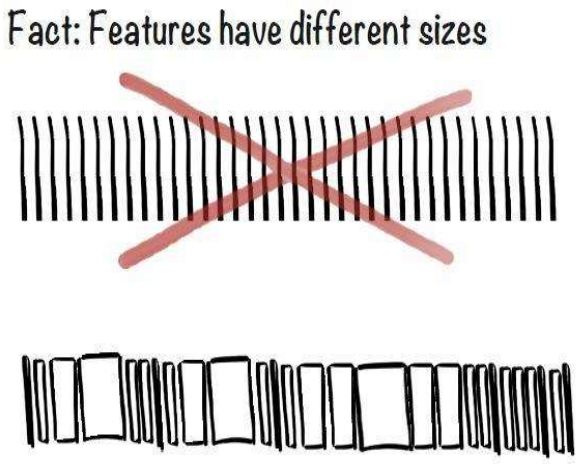
➢**User Story:** Müşteri, son kullanıcı veya ürün sahibi için değerli olan ve anlam ifade eden genellikle fonksiyonel özelliklerin belirtildiği ifadelerdir.



6. Scrum Modeli

- Örnek User Story: Online alışveriş yapan biri olarak, alışverişe daha sonra devam edebileyim diye, alışveriş kartımın kaydedilmesini istiyorum.

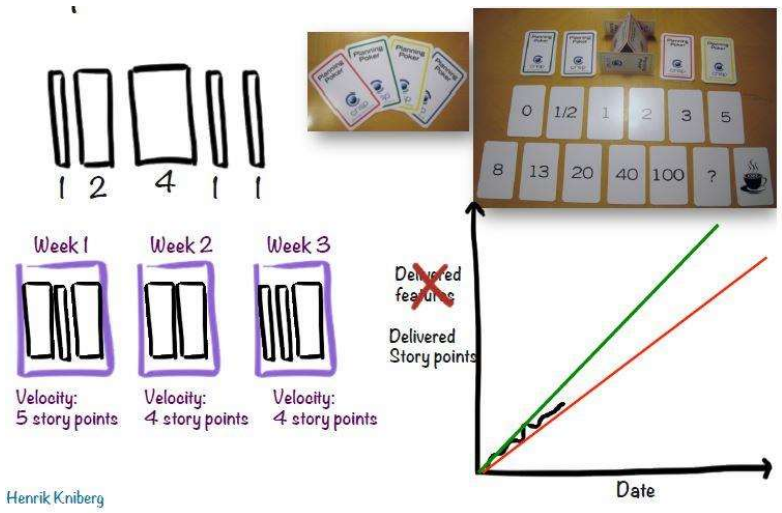
6. Scrum Modeli



- Her bir user story farklı bir boyuttadır. Somut olarak bakarsak, bir projedeki her bir gereksinim için gereken iş gücü ve zaman aynı değildir. Bu sebeple ürün backlogları sprintlere bölünürken, user storylerin boyut ve öncelikleri göz önünde bulundurulur. Örneğin bir sprint 3 user story içerirken diğeri daha küçük boyutlarda 5 user story içerebilir. Peki boyutları nasıl belirleyeceğiz?

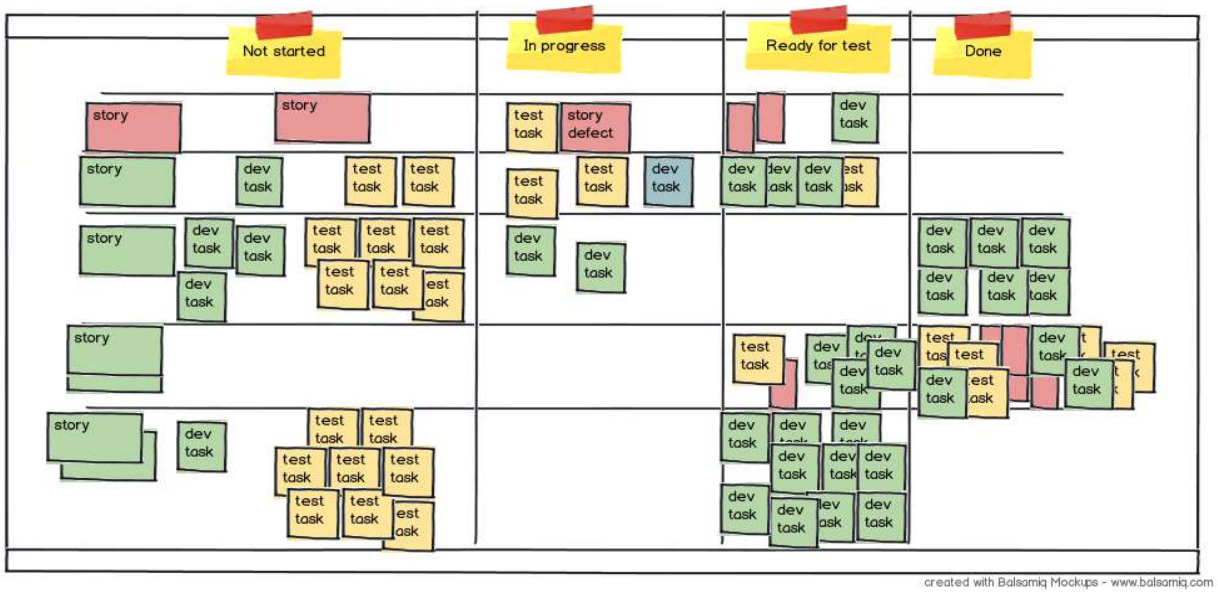
İş göründüğü gibi değil!

6. Scrum Modeli



- **Poker Kartları:** Scrum takım üyeleri bir araya gelir. Scrum master bir user story okur. Takımdaki her bir üye user story için uygun gördüğü poker kartlarından birini seçer. Herkes kartları seçtikten sonra tüm kartlar açılır ve değerlendirilir. Böylece herkesin ortak görüşü sonunda user story'lerin büyüklüğü belirlenir.

6. Scrum Modeli



Scrum Board

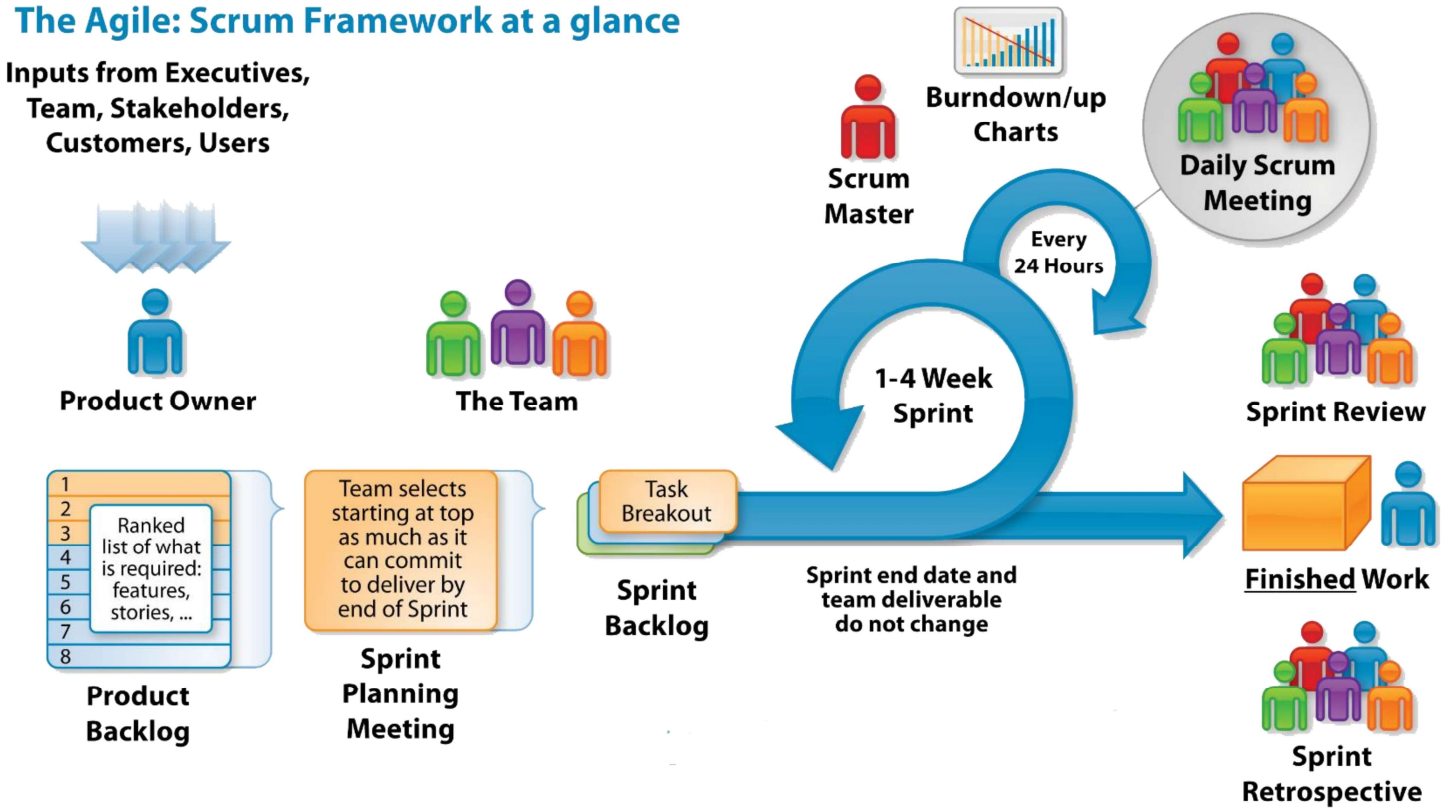
6. Scrum Modeli

SCRUM	XP (EXTREME PROGRAMMİNG)
➢Sprint (2 hafta-1 ay) ➢Sprintler en son halini aldıktan, toplantı yapıldıktan sonra değişmez. ➢Özellikler geliştiriciler tarafından derecelendirilir. ➢Herhangi bir mühendislik pratiği tanımlamaz.	➢Sprint (1 yada 2 hafta) ➢Sprintler değişebilir. ➢Özellikler ürün sahibi tarafından derecelendirilir. ➢Mühendislik pratikleri tanımlar. Eşli programlama, otomatik test, basit dizayn vs.

Agile is...

Early delivery of business value

Less bureaucracy



Çalışma Soruları

1. Agile modelinin geliştirilmesine neden ihtiyaç duyulmuştur? Açıklayınız.

2. Agile takımları kimlerden oluşur ve iş paylaşımları nasıldır?

3. Agile modelinin diğer modellerden farkları nelerdir? Üzerinde durduğu temel noktalar nelerdir?

4. User story, backlog ve sprint kavramlarını açıklayınız.

5. Agile ile geliştirilmiş büyük yazılım projelerine örnekler veriniz.

6. Scrum haricindeki diğer çevik yazılım geliştirme yöntemlerini açıklayınız.

Kaynaklar

[1] Martin, Micah, and Robert C. Martin. *Agile principles, patterns, and practices in C#*. Pearson Education, 2006.

[2] Kniberg, Henrik. "Scrum and XP from the Trenches." *Lulu. com* (2007).

[3] Kniberg, Henrik. " What is agile" , 2013

[4] http://en.wikipedia.org/wiki/Agile_software_development

[5] http://www.mshowto.org/microsoft-visual-studio-team-foundation-server-nedir.html

[6] www.kurumsaljava.com/download/10/

[7]https://www.tubitak.gov.tr/tr/destekler/akademik/uygulamalar-ve-yonergeler/icerik-proje-performans-odulu-ppo-uygulamasi

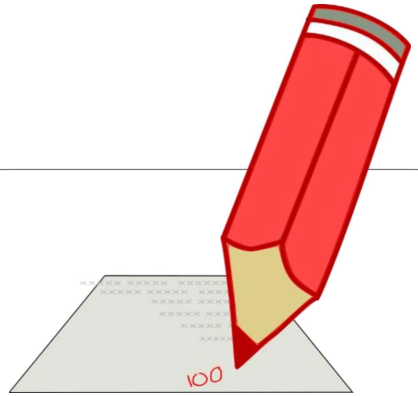
[8] http://images.slideplayer.biz.tr/8/2395426/slides/slide_27.jpg

[9] http://www.bayramucuncu.com/wp-content/uploads/2013/04/Ads%C4%B1z.png

[10] http://antasya.com/Images/is-alanlarimiz/yazilim-sistemleri/agile-method.png

[11] http://volkansel.com/wp-content/uploads/2014/07/agile-scrum.jpg

Ödev



Sorularınız