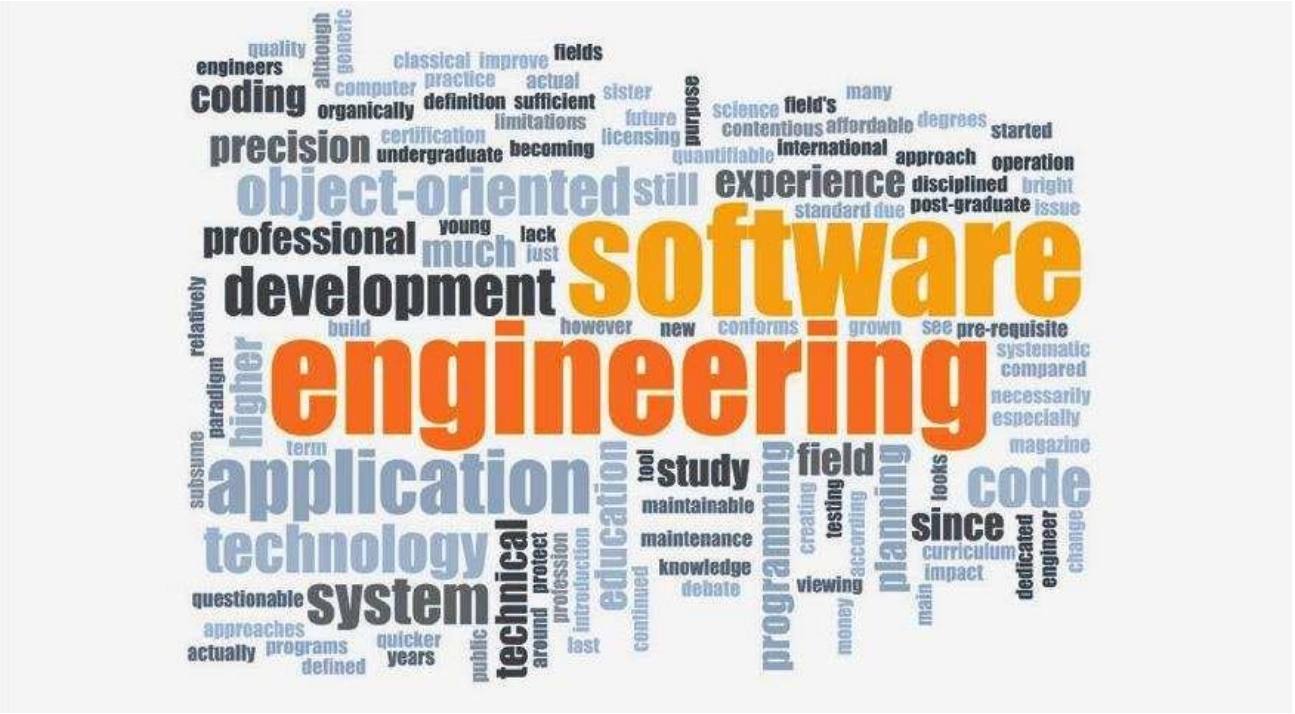
YAZILIM GELİŞTİRME YAKLAŞIMLARI

GİRİŞ:

- Bilindiği gibi 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.

- Bir yazılım ürünü geliştirme süreci; yazılım yaşam döngüsü olarak adlandırılmakta; **çözümleme**, **tasarım**, **gerçekleştirme**, **sınama** ve **bakım** aşamalarından oluşmaktadır. Yazılım yaşam döngüsü temel olarak; yazılımın nasıl geliştirileceğinin, karakteristiğinin bir tanımlanması ya da tasvir edilmesi demektir. Yazılım yaşam döngüsünün adımlarının uygulanış biçimini, seçilen yazılım geliştirme modeli belirlemektedir.

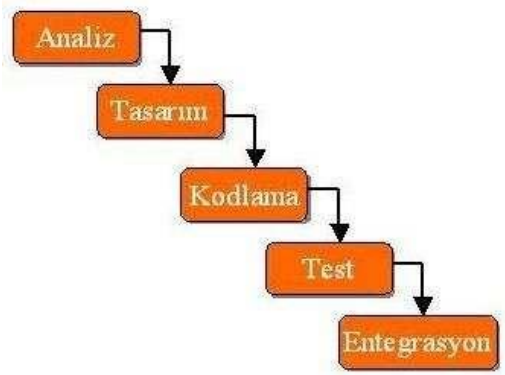
- Yazılım geliştirme modeli; yazılımın gerçekleştirilebilmesi için gerekli stratejiyi ifade eder ki bu strateji bir dizi aktiviteyi, objeyi, dönüşümü ve olayları içermektedir. Ancak seçilen her modelin kendine özgü avantaj ve dezavantajları bulunmaktadır. Dikkat edilmesi gereken nokta çözüm istenen ürüne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

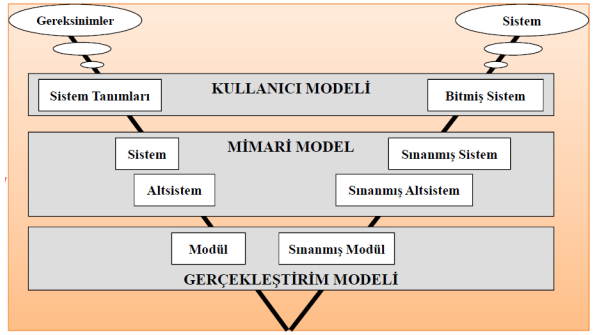
Bir yazılım geliştirme metodolojisi aşağıdaki adımlardan meydana gelir;

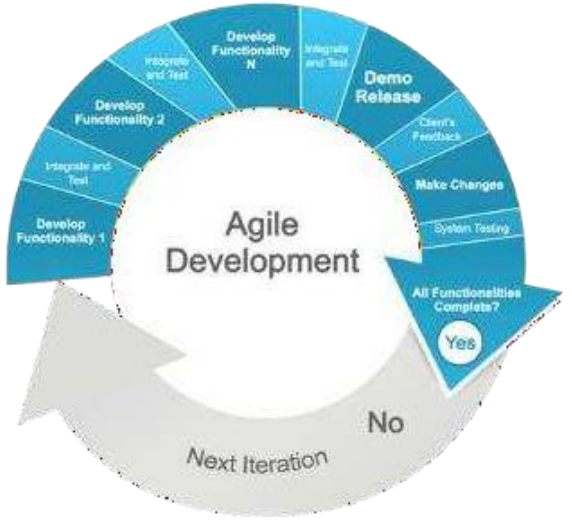
- Yazılım geliştirme süreci yaklaşımıyla bir yazılım geliştirme felsefesi
- Yazılım geliştirme sürecine destek verecek araçlar, modeller ve yöntemler.

Bu özellikler çoğunlukla metodolojiyi geliştiren, destek veren ve onu ilerleten organizasyon tiplerine bağımlıdır.

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







Tarihçe:

- En eski yazılım geliştirme araçlarından biri, kökleri 1920'lere dayanan **flowchart(akış diyagramları)**'lardır. Yazılım geliştirme metodolojisi 1960'lara kadar tam olarak ortaya çıkmamıştır.

Spesifik Yazılım Geliştirme Metodolojileri :

- **1970'lerde..**

Yapısal programlama

- **1980'lerde..**

Yapısal sistem analizi ve tasarımı metodolojisi

- **1990'larda..**

Nesne yönelimli programlama

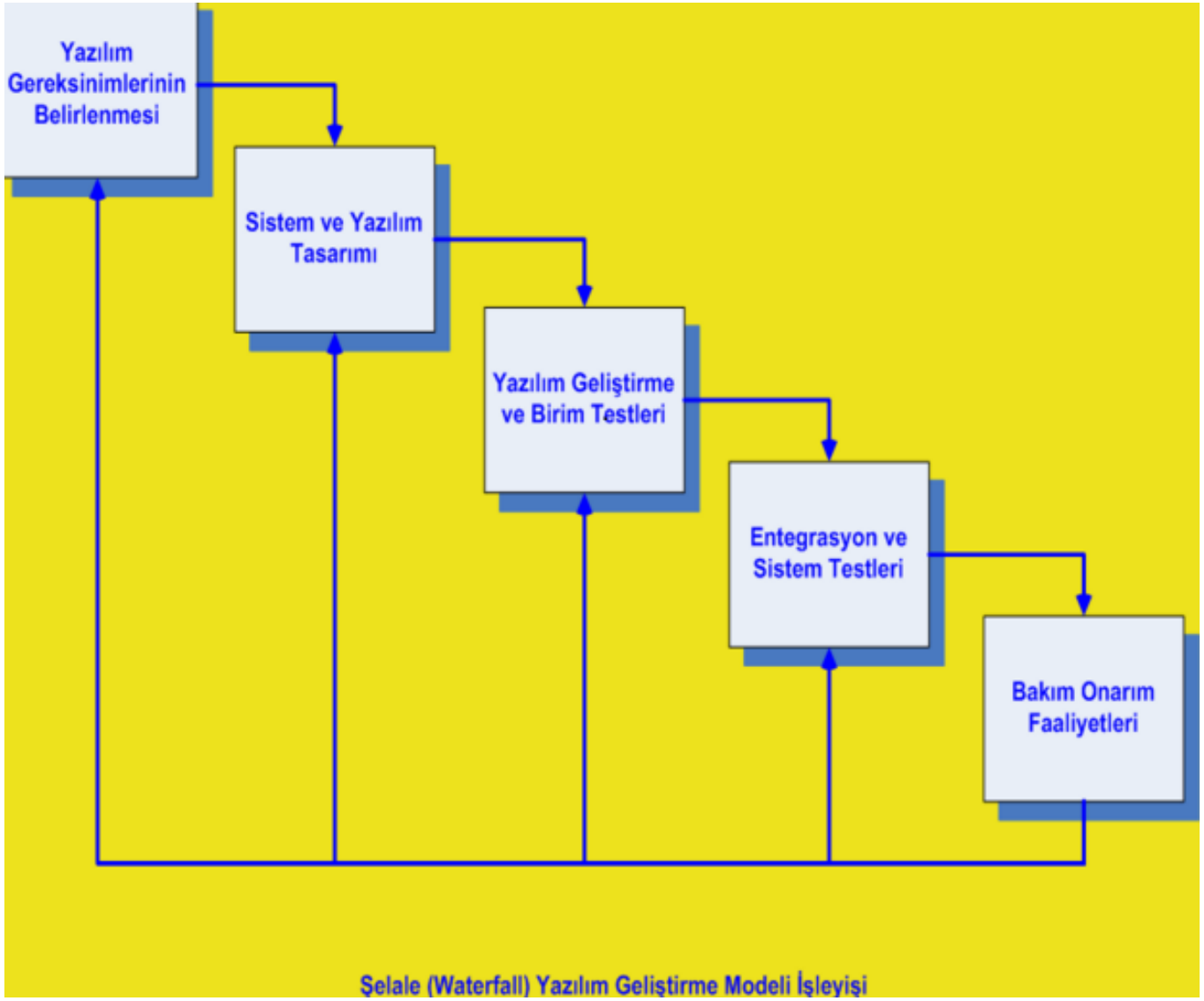
Rapid Application Development(Hızlı Uygulama Geliştirme)

Scrum Development

Team Software Process(Takım Yazılım Süreci)

1-)Şelale Modeli.

- Yakın zamana kadar popüler bir yöntemdi.
- Geleneksel yazılım geliştirme modeli olarak da bilinir.
- Yazılım aşamaları en az bir kez tekrarlanır.
- Çok iyi tanımlanmış ve üretimi az zaman gerektiren projeler için uygun bir model olmakla birlikte…



Şelale (Waterfall) Yazılım Geliştirme Modeli İşleyişi

- …günümüzde kullanımı gittikçe azalmaktadır.
- Dökümantasyonu ayrı bir süreç olarak değil, üretimin doğal bir parçası olarak görür.
- Ayrıca, bu modelde aşamalar arasındaki geri dönüşlerin nasıl olacağı da tanımlıdır.
- Ancak, şelale modelinin kullanımında dikkat edilmesi gereken önemli hususlar vardır.
- **Bunlardan en önemlisi,** analiz aşamasında müşteri ve sistem gereksinimleri en ince ayrıntısına kadar belirlemek tasarım aşamasında gereksinimleri karşılayacak çalışma yapmak.. Dolayısıyla ekipler en fazla bu iki alanda efor harcayacaklar. Tüm bu efor ve detaylı çalışmalara rağmen özellikle uzun zamana yayılan projelerde gereksinimlerin değişecek olması kaçınılmazdır. Kodlama veya test aşamalarında olabilecek bu değişikliklerin sisteme/yazılıma yansıtılması maliyeti ise çok yüksektir.

- Kullanıcı, yazılımın geliştirilme aşamasında sürecin içerisinde genellikle yer almaz ve bu durum yazılım tamamlandıktan sonra geri dönüşleri arttırabilir. Maliyeti artıran bir durumdur.

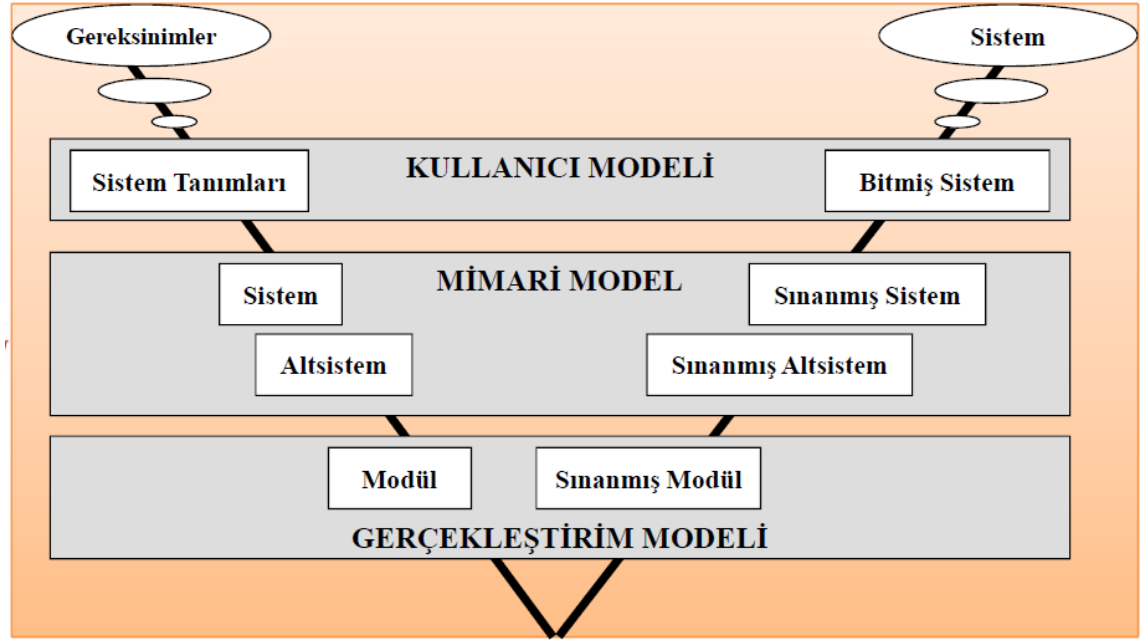
- Tasarım aşamasında fark edilen hata ve eksiklikler küçük bir maliyet ile giderilebilir. Ancak, bu hata ve/veya eksiklikler entegrasyon veya bakım aşamalarında fark edilirse bunları gidermenin maliyeti 50-200 kat artacaktır.

2-)V Modeli:

- isterlerin iyi tanımlandığı,
- belirsizliklerin az olduğu,
- aşamalar halinde ilerlenmesi erken

projelerde V modeli iyi sonuçlar verir.

Sistem geçerleme sırasında karşılaşılan sorunu gidermek için sistem isterleri çözümlemesinin tekrarlanması gerekir.

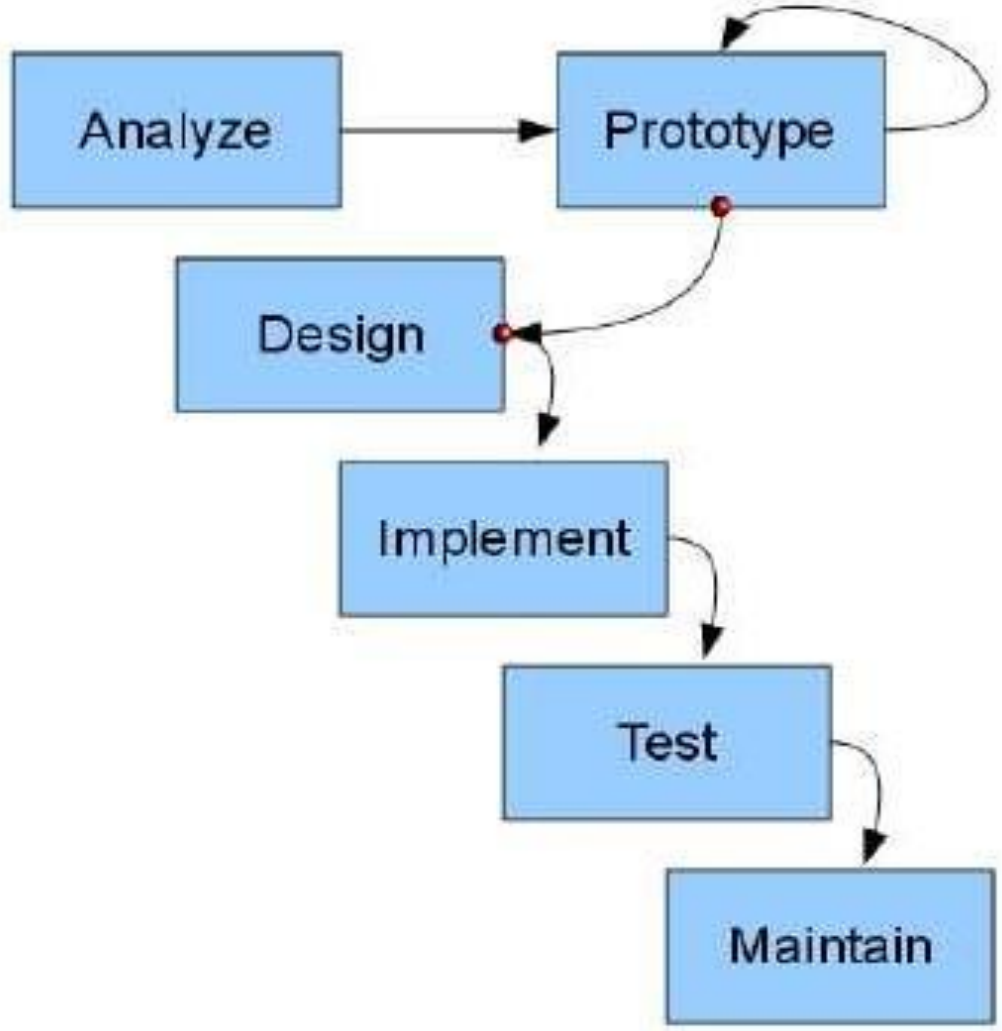


- Şelale modelin gelişmiş hali olarak düşünülebilir.
- Modelin sol tarafı proje tanımı(**doğrulama**), sağ tarafı test kısmını(**geçerli kılma**) oluşturur.

doğrulama	**geçerli kılma**
gereksinim analizi	birim test etme
sistem tasarımı	entegresyon test etme
mimari tasarım	sistem test etme
modül tasarımı	kullanıcı kabul test etme

3-)Prototip Modelleme

- 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, yadaiptal edilip en baştan yapılabilir.



- Yazılım parça parça geliştirilir.
- Tüketici tasarım aşamasında katılım sağlayabilirlerse bu faydalı olur.
- Bilgi toplama ile başlar, sonra prototipler üretilir ve tüketicilere değerlendirme için verilir.
- Müşteri üründen memnun olduğunda, değerlendirme sürecini bütün sistemin ihtiyaçlarını karşılamaları sağlamak üzere prototipi genişletmek için sürekli kullanılabilir ya da prototip tamamen kaldırılabilir.

4-)Spiral Model

Temel olarak 4 ana bölüm iç

- Planlama-risk yönetimi-üretim-kullanıcı değerlendirme
- **Planlama**: Amaç belirlenir. Alternatifler belirlenir. Kısıtlar belirlenir.
- **Risk Yönetimi:** Alternatifler değerlendirilir.

  Risk analizi yapılır.

- **Üretim:**

Geliştirme yapılır.

Bir sonraki ürün belirlenir.

- **Kullanıcı Değerlendirmeleri**

Bir sonraki aşama planlanır.

Kullanıcı değerlendirmeleri alınır.

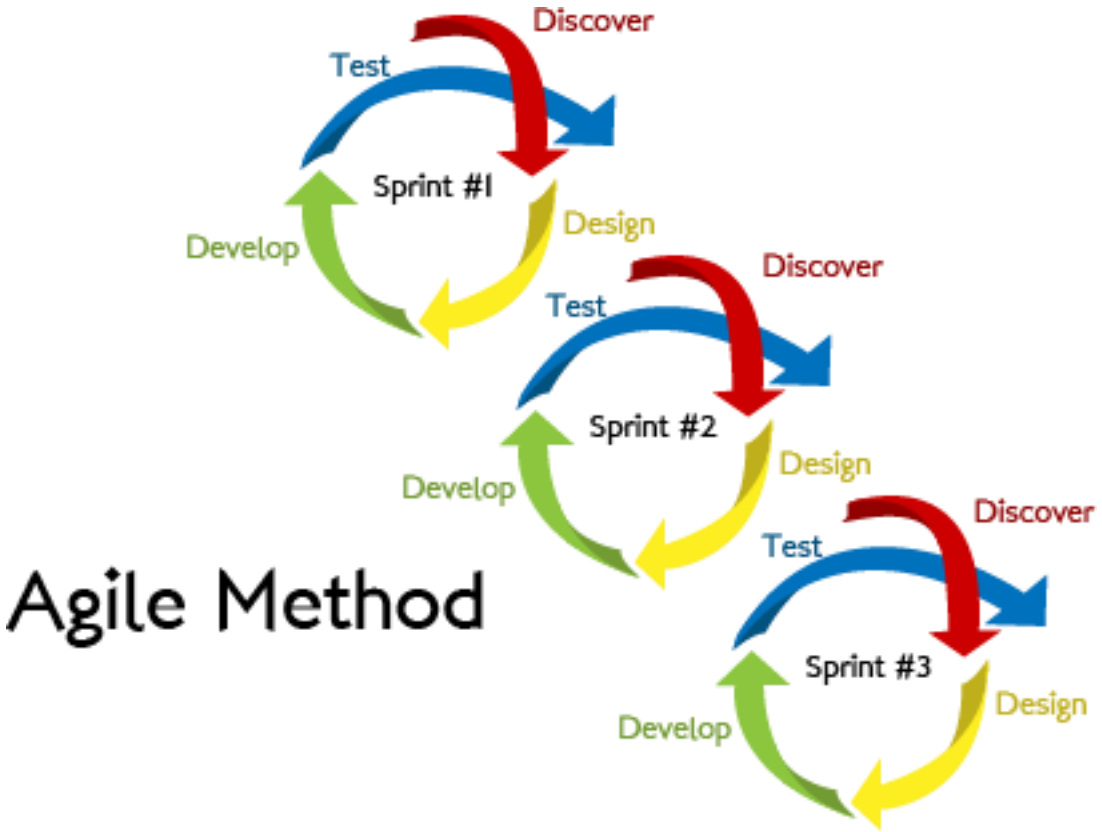
- Model risk analizi ve prototip üretme üzerine kurulmuştur. Her döngü öncesi içinde bulunulan fazın risk analizi yapılır ve o faz için planlanmış olan prototip geliştirilir. Her döngünün sonunda yeniden planlama yapılarak hedefler alternatifler ve kısıtlamalar belirlenir.
- Bu model önceden geliştirilmiş yazılım bileşenlerinin yeniden kullanıldığı projeler için çok uygundur.

- Modelin en önemli getirisi; her döngünün başında risk analizi yapıldığı için zaman ve maliyet bileşenlerinin kolay tahmin edilmesidir.

- Ancak model; küçük yazılım projelerinde kullanılmasının uygun olmaması, modeli uygulayanların bu konuda tecrübeli olması gerekliliği, risk analizi olgusuna dayandığından zorluklar taşıması gibi dezavantajlarda taşımaktadır.

5-)Çevik Geliştirme

- Yetersiz olduğu düşünülen yazılım geliştirilme modellerine bir alternatif olarak gelişmeye başlamıştır.
- Ardışıl bir tasarım yerine, geliştiriciler basitleştirilmiş küçük modüller üzerinde çalışmaya başlarlar. Modüller bir hafta veya bir aylık kısa dönemler içerisinde yapılır ve her bir dönemin sonunda proje öncelikleri değerlendirilerek testler gerçekleştirilir.



- Bu kısa dönemler yazılım hatalarının keşfedilmelerini sağlarken, aynı zamanda kullanıcıdan gelen geri bildirimlerin de modül içerisine eklenmesine olanak sağlar.

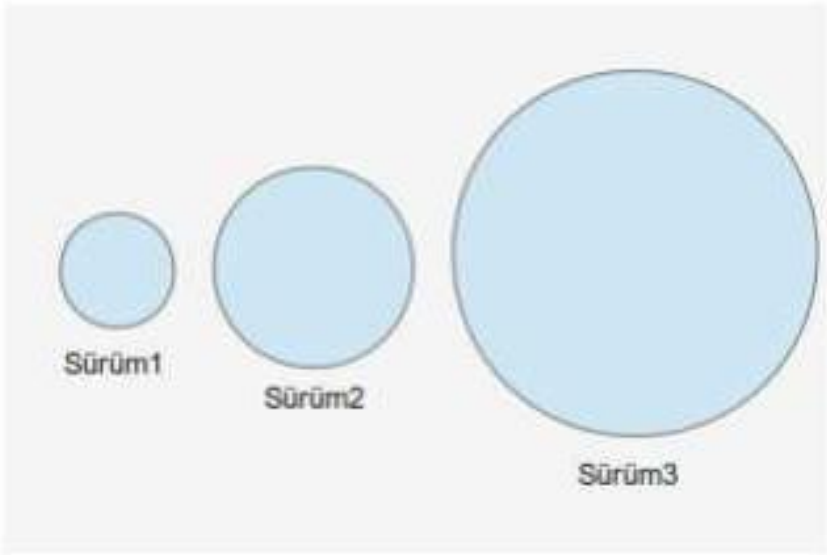
model bize;

- Modüler olarak geliştirilen yazılım, kullanıcılara kısa dönemler içinde teslim edilerek müşteri memnuniyeti,
- Kullanıcılar, geliştiriciler ve test edicileri arasında sürekli işbirliği ,
- Değişen koşullara karşı projenin adaptosyon sağlaması,
- Projenin ilerleyen aşamalarında ihtiyaç duyulan gereksinimler eklenebilirliği,

gibi avantajlar sağlar.

6-)Evrimsel Geliştirme

Her evrimde geliştirilen ürünler uygulama alanında tam işlevselliğine sahiptir. Modelin uygulamadaki başarısı ilk evrimde ortaya çıkan ürüne bağlıdır.



7-)Evrimsel Prototipleme

Evrimsel gelistirmeden biraz farklı olarak, her evrimde ortaya bir prototip yani ön ürün çıkarılır. Ön ürünün uygulama alanında denenmesinden sonra, kullanıcı girdileri alınarak bir baska ön ürün olusturulur ve denemeye sunulur. Bu sekilde gelistirmeye devam edilerek ürün son haline getirilir ve sonra kullanıma sunulur.

8-) Artımlı Geliştirme

- Önce en temel isterlere göre çekirdek yapıda bir ürün geliştirilir.
- Bu ürün asıl sistemin temel işlevlerini yerine getirebilecek durumdadır, ancak bazı işlevleri eksiktir.
- Her sürüm bir öncekinden daha fazla işlevselliğe sahiptir.
- Tüm isterleri tanımlı bir yazılımın dört sürüm halinde gerçekleşmesi artımlı olarak gerçekleşmektedir.
- Bu model evrimsel gelistirmeye benzemekle beraber en büyük farkı, sürümlerin tam islevsellik içermemesidir.

9-) Araştırmaya Dayalı Geliştirme:

- Araştırma projeleri genellikle tam belirsizlik ile baslar.
- Belirli bir konuda deneyim kazanmak amacıyla geliştirme yapılabilir.
- Araştırmaya dayalı olduğu için sonuçları hakkında kesin bir şey söylenemez.
- Bu araştırmalar asıl sistem geliştirme için kullanılacak alt yapıyı oluşturacaktır.

10-)Uç Programlama(Extreme Programming – XP)

- Değişen müşteri istekleri, ürünün yaşam çevrimi sırasında mutlaka
- karşılanır. XP ile yazılım projesi dört ilkeye bağlı olarak geliştirilir;

1. İletişim
2. Basitlik
3. Geri Besleme
4. Cesaret

- XP, geliştirme sırasında isterleri sürekli değisen problem alanları için Agile(Çevik) yazılım geliştirme yöntemidir.

11-) Gelişigüzel Geliştirme:

- Yazılımda ne destek vardır, ne de disiplin.

SONUÇ:

- Yazılım projesi diğer proje türlerinde çok daha farklı yapıya sahiptir. Gelişen teknolojinin hızlı değişimi ve tamamen insan beyin emeğini gerektiren yapısı ve öngörülemeyen unsurların çok olmasından dolayı yazılım projeleri ve bunların yönetimine ait süreçlerin iyi seçilmesi ve yönetilmesi gerekmektedir. Dünyada gerçekleştirilen yazılım ve teknoloji tabanlı projelerin büyük çoğunluğu istenen hedeflere ulaşamamaktadır. Bu durum teknolojinin ve proje sahibinin isteklerinin proje sürecinde değişmesi, proje başlangıcındaki isteklerini tam olarak belirleyememesinden dolayı yanlış analizlerin oluşturulmasından kaynaklanmaktadır. İyi bir sonuç elde etmek için doğru modülün seçilmesi gerekir.