YAZILIM YAŞAM DÖNGÜSÜ

* Yazılım da aslında bir üründür ve ürünlerin bir yaşam süreci vardır.
* Yazılım yaşam döngüsü tek yönlü veya doğrusal değildir ve birkaç temel aşamadan oluşur.
* Yazılım ürününü geliştirmek için teorik olarak ne yapılacağının tanımlanmasını ifade eder.
* Yaşam-döngü modeli içerisindeki safhaların sayısı ve niteliği kullanılan modele göre değişir.
* Yani yazılım projelerinin analiz ve planlanmasından başlayarak kullanımı boyunca süren bir döngüdür diyebiliriz.Bu aşamalar şunlardır:

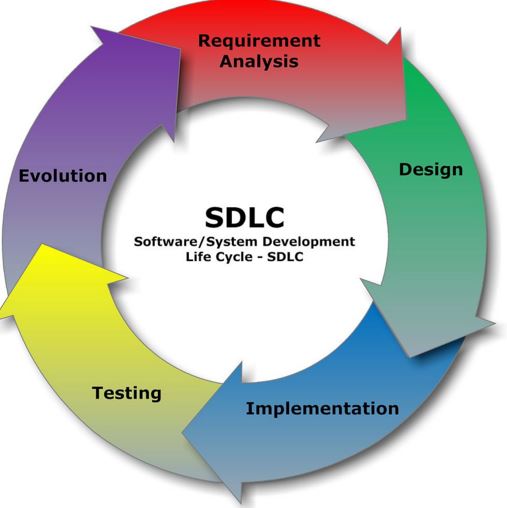
1.Planlama :İlk adımımız, ne istenildiğinin planlanması olarak düşünülmelidir. Bu plan aşaması aslında proje yönetimlerinde en önemli aşamasıdır. Diğer adımların da başlangıç aşamasıdır. İşin projelendirildiği, fikrin ortaya çıkarıldığı ve fikrin tartışıldığı aşamadır.

2. Tanımlama: İstenilen fikrin ne olduğu ve temel tanımlar üzerinde konuşulacak kavramların tanımlandığı aşamadır.Analiz aşaması olarak da düşünülebilir. Problemin tanımlandığı veya yaşam döngüsünün tanımlandığı sistemin veyahut yazılımın tanımlandığı aşamadır. Tanım aşamasında projede nelerin istenildiği ile ilgili analiz çalışmaları da yapılabilir.

3. Tasarım :Yazılımımızın veya sistemimizin tasarımları yapılır. Projeleri çizilir. Planlama ve tanımlaya göre bir tasarım çizilir. Kararlar verilir, seçimler yapılır.

4. Gerçekleştirme :Bu aşama, yazılım projeleri için kodlama olarak düşünülebilir veya sistemin yaşatılmaya başlandığı ilk örneklerinin çıkmaya başlandığı aşama olarak düşünülebilir.

5. Bakım: Test etme ve kurulum aşamasından sonra gerçek ortama kurulunan yazılımın, hata giderme, güncelleme gibi işlemlerin yapıldığı aşamadır.Bu aşama yazılım tüm yaşamı boyunca sürer.

. 

YAZILIM YAŞAM DÖGÜSÜ MODELLERİ

1.GELİŞİGÜZEL MODEL

* Herhangi bir model veya yöntem gözetmeksizin yapılır.
* İlgi duyan ve geliştiren kişiye bağımlıdır. Takip edilebilirliği ve bakımı oldukça zordur.

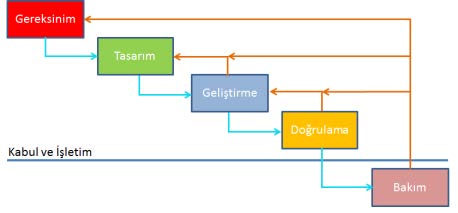
2.BAROK MODELİ

* Bu model yaşam döngüsü başlıca adımlarının doğrusal bir biçimde geliştirildiği modeldir.
* Adımlar arası geri dönüşlerin ne türlü yapılacağı da tanımlanmamıştır.
* En kötüsü de gerçekleştirim evresine daha çok ağırlık veren bir model olduğundan bugün kullanımı tavsiye edilmemektedir.

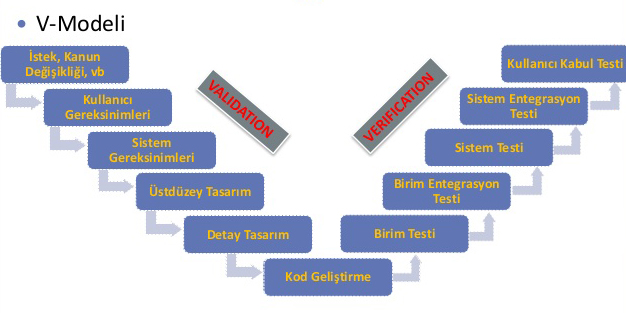
3.ÇAĞLAYAN(ŞELALE) MODELİ

Öncelikle gereksinimlerin tanımlanması yapılır.

* Sonrasında sistem ve yazılım tasarımı gerçekleştirilir.
* Beraberinde kodlama ve parça test etme etapları gerçekleştirilir.
* Yapıyı birleştirme ve sistemi test etme yapıldıktan sonra da sistemin bakım ve idamesi sağlanır.
* Bu modelin bazı sorunları vardır.
* Gerçek hayattaki projeler genelde tekrar gerektirir.
* Bütünde yazılımın kullanıcıya ulaşma zamanı uzundur.
* İhtiyaç tanımlamaları çoğu kez açık bir şekilde yapılamadığından ötürü, hataların düzeltilme ve eksikleri giderme maliyetleri artırır.
* Üst düzeydekilerin ürünü görme müddetinin uzun olması, projenin bitmeyeceği ve devamlı gider merkezi boyutuna geldiği düşüncesini artırmaktadır.

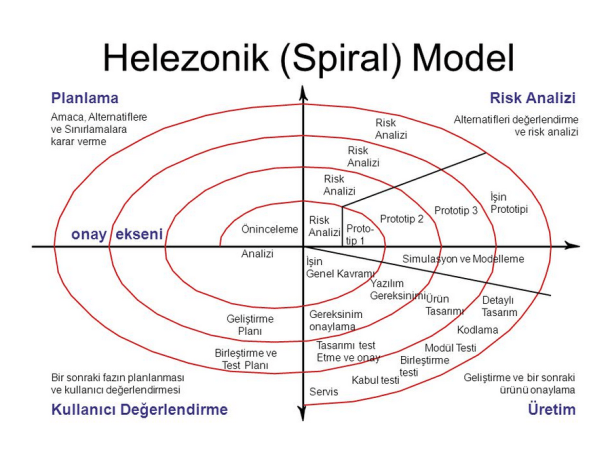


4.V SÜREÇ MODELİ

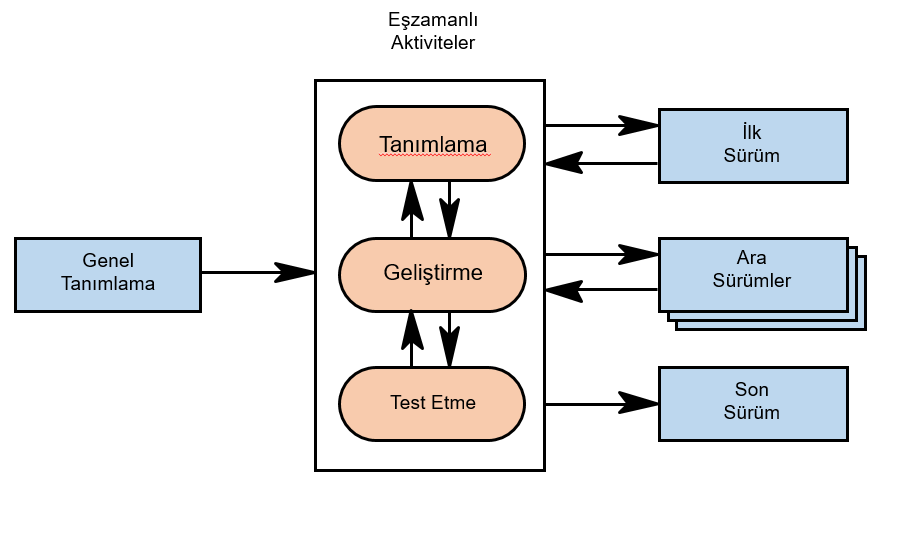
* V-model(yazılım geliştirme) şelale(waterfall) modelinin gelişmiş hali olarak düşünülebilecek bir yazılım geliştirme süreci sunar.
* Doğrusal bir yönde ilerlemek yerine, süreç adımları kodlama evresinden sonra yukarıya doğru eğim alır ve tipik V şeklini oluşturur.
* V-Model geliştirme yaşam çevriminin her bir evresi arasındaki ilişkileri gösterir.
* Yatay ve dikey açılar zaman veya projenin tamamlanabilirliğini ve soyut seviyeyi gösterir.

5.HELEZONİK(SPİRAL) MODELİ

* Modelin geçirdiği safhaları sıralamak gerekirse; önce Planlama gelir. Ardından Risk Analizi ve Üretimi saymalıyız. Son olarak da Kullanıcı Değerlendirmesi olarak bitirilmelidir.
* Bu modelin avantajları vardır.
* Kullanıcı Katkısı en ön plandadır.
* Yönetici Bakışı çok önemlidir.
* Bir de Yazılım Geliştirici (Mühendis) Bakışı gibi avantajlar sağlar.
* Helezonik Modeli ifade için önce söylenecek şey; Risk Analizi Olgusu ön plana çıkmıştır.
* Her bir döngü bir fazı tasvir eder.
* Direkt tanımlama, tasarım ve saire gibi bir faz yoktur.
* Artımsal bir yinelemeli yaklaşım vardır. Bu modelde Prototip yaklaşımı mevcuttur.



6.EVRİMSEL GELİŞTİRME SÜREÇ MODELİ



* Bu model İlk tam ölçekli olan modeldir.
* Başarısı ilk evrimin başarısına bağımlı haldedir.
* Tüm aşamalarda üretilen ürünler, üretildikleri yer için tam fonksiyonelliği içermektedirler
* .Zayıf yanıysa değişiklik denetimine sahip olmamasıdır.
* Modelin yapılandırma, kalite, sürüm ve değişiklik yönetimi biraz problemlidir.

7.ARTIRIMSAL GELİŞTİRME SÜREÇ MODELİ

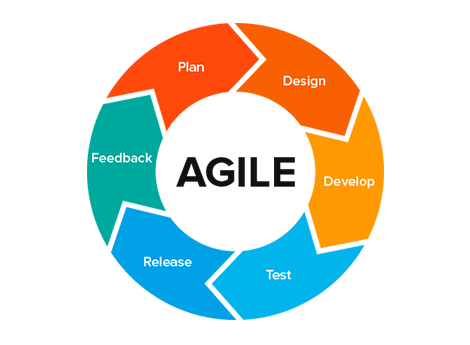
Üretilen her bir yazılım sürümü diğerini kapsayacak ve giderek fazlalaşan adette işlev içerecek şekilde modernleştirilir.

8.ARAŞTIRMA TABANLI SÜREÇ MODELİ

Model Yap-at ilk örnek olarak da bilinir. Bunun araştırma ortamları tümüyle belirsizlik üzerine çalışan koşullardır. Modelde yapılan işlerden elde edilecek neticeler belirgin değildir.

9.AGİLE(ÇEVİK)MODEL

Çevik yazılım geliştirme; yinelemeli (iterational) olarak yazılım geliştirme sürecinin aktivitelerinin uygulanması ve tedrici (incremental) olarak yazılım ürününün doğru bir şekilde, değişime açık, Agile(Çevik) Birliği’nin oluşturduğu değerler ışığında prensiplerini uygulayarak geliştirilmesidir.



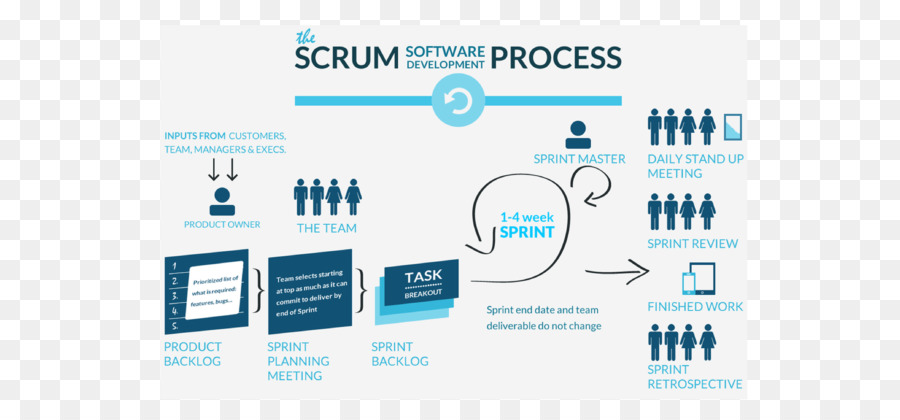
SCRUM

* Başarılı proje yönetimi tekniklerinden biri de Scrum’dur.
* Scrum, Agile proje yönetimi metotlarından biridir.
* Karmaşık yazılım projelerinin başarı oranını yükseltmek için oluşturulmuştur.
* En yaygın kullanılan proje yönetim metotlarından biridir
* .Scrum, projeyi küçük parçalara ayırarak yönetmeyi önerir. Bu sayede her parça tek tek incelenip gerekli değişiklikler yapılabilir. Bir parçada olan sorun daha kolay bulunabilir.
* Scrum’a göre proje yönetilirken müşteri ile daima iletişim halinde olunması gerekir.
* Scrum’un projeyi küçük parçalara ayırmasındaki en temel sebeplerden birisi, müşterilerin isteklerine göre projenin şekillendirilmesini kolaylaştırmaktır.
* Scrum 3 temel üzerine oturtulmuştur:

1. Şeffaflık
2. Kontrol edilebilme
3. Uyumluluk

SCRUM GÜNÜMÜZDE NEDEN POPÜLER?

* Scrum günümüzde en çok kullanılan yazılım geliştirme yöntemidir. Hatta sadece yazılım geliştirmede değil birçok sistemin geliştirilmesinde de kullanılır. Bunun nedenleri ise; zamandan ve paradan büyük ölçüde tasarruf edilmesi, yüksek teknolojiler ve son gelişmelere kolaylıkla uyum sağlayabilmesi, karmaşık görülen ve gereksinimleri tam belirlenmemiş projeler için ideal olması, ekip içi iletişimin yüksek tutulması ve bununla beraber hataların erken fark edilip düzeltilmesi, kullanıcıdan sürekli geri bildirim gerektirmesi ve bununla beraber sorunların azalması, diğer yazılım geliştirme metodolojileri gibi yinelemeli olması, değişen gereksinimlere hızlı bir şekilde tepki vermesi gibi nedenler örnek olarak verilebilir.



MODELLERİ KARŞILAŞTIRALIM

Çağlayan Modeli,V Modeli,Evrimsel Modeli karşılaştıralım:

Çağlayan Modeli V Modeli Evrimsel Modeli

Zaman çok uzun uzun uzun

Uygulama kolay kolay kolay

Basitlik basit orta karmaşık

Başarı düşük orta -

Bakım düşük düşük düşük

Maliyet maliyetli maliyetli düşük

YARARLANDIĞIM KAYNAKLAR:

<https://medium.com/@denizkilinc/yaz%C4%B1l%C4%B1m-ya%C5%9Fam-d%C3%B6ng%C3%BCs%C3%BC-temel-a%C5%9Famalar%C4%B1-software-development-life-cycle-core-processes-197a4b503696>

<https://www.codex.com.tr/yazilim-gelistirme-modelleri>

<https://fikirjeneratoru.com/yazilim-proje-yonetimi-yontemleri/>