**Yazılım Yaşam Döngüsü**

Yazılım yaşam döngüsü, bir yazılım ürününün oluşturulması ve bakımı da dahil olmak üzere yapılan adımlar topluluğuna denir. Yazılım yaşam döngüsü doğrusal veya tek yönlü değildir bazı temel aşamalardan oluşur. Bu aşamalar; Planlama, Analiz, Tasarım, Gerçekleştirme, Bakımdır. Kısaca bu aşamalardan bahsedecek olursak.

**Planlama:** Projenin planlanması ve görev dağılımı yapılır.

**Analiz:** Projenin ne kadar süreceği ve riskleri belirlenir.

**Tasarım:** Projenin nasıl sürdürleceği tasarlanır.

**Gerçekleştirme:** Projenin kodlandığı ve test edildiği kısımdır.

**Bakım:** Proje sunulduktan sonra güncelleme, onarım gibi olayların yapıldığı kısımdır.

Yazılım yaşam döngüsünde birden fazla model vardır. Birden fazla olmasının sebebi ise projelerin büyüklük farklılıkları ve hazırlandığı kişi ve kitlelerin farklılığı gibi sebeplerdir. Yazılım sektöründe en çok kullanılan modeller ;

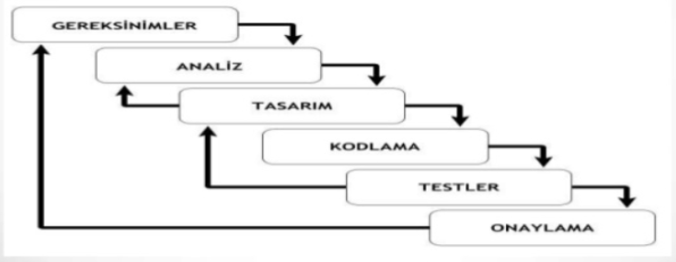
**1- Gelişigüzel Model**

Bu yöntem aslında bir model değildir. Bu geliştirmede belli bir model ya da yöntem bulunmaz. 1960’larda kullanılmıştır. Kişiye bağlı olduğu için bakımı zordur.

**2- Barok Modeli**

Bu modelde, dökümantasyon ayrı bir süreç olarak ele alınır. Bu günümüzde kullandıklarımızdan farklıdır. Bu yüzden günümüzde uygulanan bir model değildir. 1970’li yıllarda ortaya çıkmıştır.

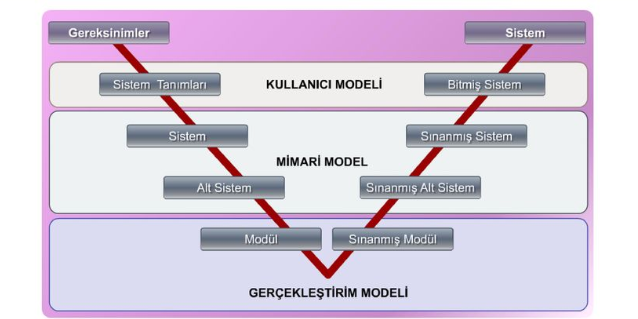
**3- Şelale (Waterfall) Modeli**

****

Bu model aslında diğer modellerin temeli niteliğindedir. Yani gelenksel bir modeldir. Bu modelde her aşama en az bir kere tekrar edilir. Bu model genelde iyi tanımlanmış ve kısa sürede bitecek projeler icin kullanılır. Bir adım bitmeden diğer adımlara geçilmez. Uzun süreli işler için uygun değildir. Kullanıcı sonucu sonda gördüğü için ortaya çıkacak bir sorun maliyeti ve harcanan zamanı yükseltir.

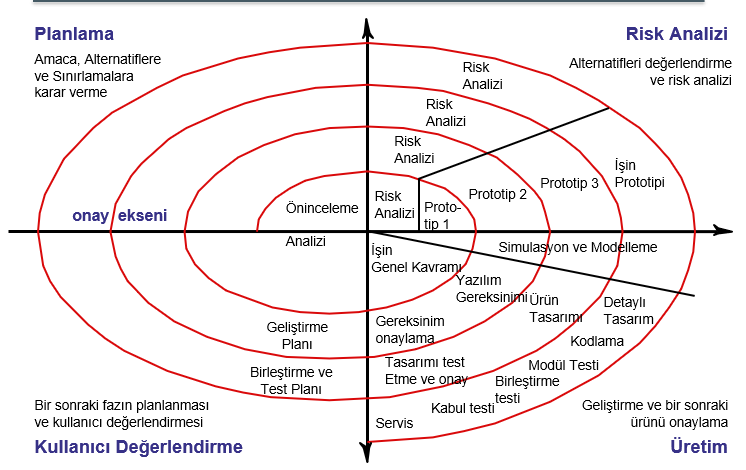
**4- V Süreç Modeli**

V modeli şelale modelinin gelişmiş hali gibidir. Bu model belirsizliğin az olduğu belirgin projelerde kullanılır. Sol taraf üretim sağ taraf sınamadır. 3 bölümden oluşmaktadır. Kullanıcı modeli, mimari modeli, gerçekleştirim modeli. Kullanıcı modelinde istekler yerine getirildikten sonra tamamlanmış hali teslim edilir. Mimari modelinde tasarım ve deneme işlemleri yapılır. Gerçekleştirim modelinde ise kodlama işlemleri yapılır.



V modelin sorunlar ise şunlardır: Aşamalar tekrarlı değildir ve risk çözümleme için alan yoktur.

**5- Helezonik (Spiral) Model**



Helezonik modelin diğerlerinden farkı risk analizinin ön planda olması ve prototip oluşturulmasıdır. Bu nedenle hatalar erken giderilebilir. Her aşamada prototip oluşturulduğu için kullanıcı her aşamada ürünü görme imkanı vardır. Bu nedenle sorunlar azalır. Bu model 4 temel aşamdan oluşur ;

**Planlama:** Her aşamada ara ürünü planlama.

**Risk analizi:** Risk araştırma belirleme çözme.

**Üretim:** Ara ürün üretilmesi.

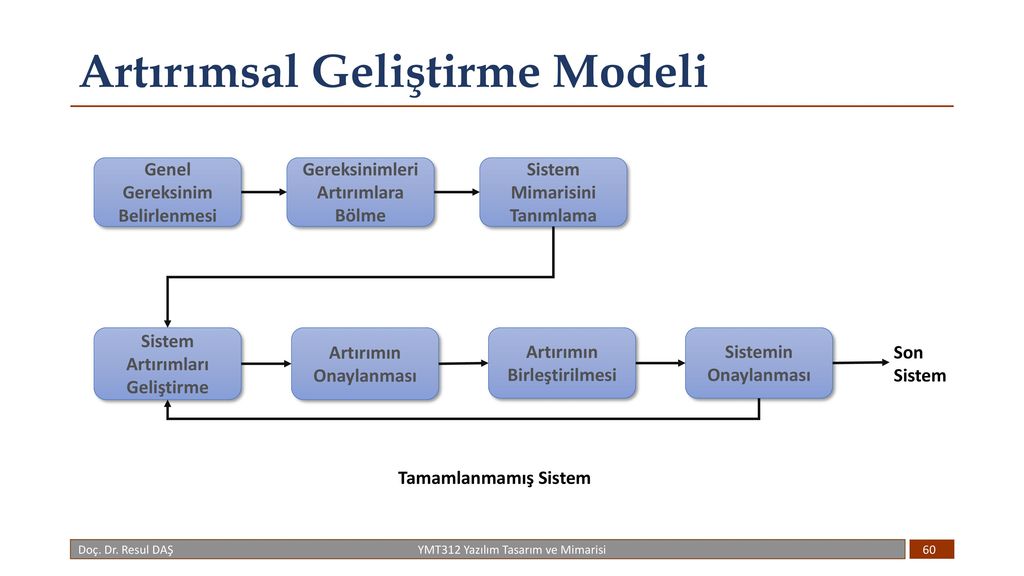
**Kullanıcı Değerlendirmesi:** Kullanıcıdan alınan geri dönütlerin değerlendirilmesi.

Bu modelin sorunları ise şunlardır: Küçük ve düşük riskli projeler için çok pahalı bir sistemi vardır. Karmaşık bir içeriğe sahiptir. Uzun sürmesidir.

**6- Artımsal Geliştirme Süreç Modeli**

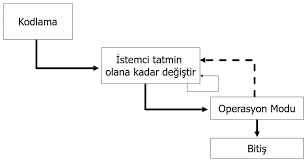
Bu modelde proje parçalara bölünür ve kullanıcının önceliğine göre sıralanır. Bu parçalarla ara ürün oluşturulur ve kullanıcı tarafından kullanılır. Ara ürünler bir öncekinin üstüne birşeyler katarak gider. Bir yandan üretim yapılırken bir yandan kullanıcı tarafından kullanılır. Bu model uzun zaman alabilecek ve eksik işlevsellikle çalışabilecek türdeki yazılımlar için uygundur. Ara ürünler geliştirilmedi büyük öneme sahiptir.

Bu modelin sorunları ise şunlardır: Her bir parçanın kendi içinde tekrar etmemesi bu nedenle bir ara ürün bitip diğeri başlayana kadar herhangi bir değişiklik yapılamaz.



**7- Kodla ve Düzelt Yaşam Döngü Modeli**

Küçük projeler için kullanılabilir. Direk ürün gerçekleştirilir ve diğerlerinden farklı olarak emeklilik safhası vardır. Büyük projeler için kullanılamaz. Bakım az vardır ve zordur. Ürün hazırlanır ve kullanıma sunulur.

****

**8-Evrimsel Geliştirme Modeli**

Bu model ilk tam ölçekli modeldir. Büyük alanlara yayılmış bir modeldir. Büyük firmalar için önerilir. Modelin başarısı ilk evrimin başarısına bağlıdır.

**Çevik Modeller**

Bu modeller başarı oranını ve ekip içi iletişimi artırmak için ortaya çıkmıştır. Bazı çevik modeller şunlardır;

**1-Extreme Programming (XP)**

Kent Beck ve arkadaşları tarafından 1996 yılında kurulmuştur.4 temelden oluşur: Basitlik, Cesaret, Geri Dönüş, İletişim.

**Basitlik:** Yapılan işin sade ve anlaşılır yapılmasını gerektirir.

**Cesaret:** Projelerin üstüne korkmadan ilerlenmelidir. Gerekirse bütün kodun silinip yeniden yazılması sağlanır.

**Geri Dönüş:** Geri dönüşlerle yapılabilecek hatalar hatalar azaltılır. Müşteri ile yazılım ekibi birbiriyle iletişim halindedir.

**İletişim:** Projelerdeki önemli sorunlardan biridir. XP bu sorunu çözmek için çalışır ve ekip içi iletişime önem verir artırılması için çalışır.

**2-SCRUM**

Büyük projeleri parçalara bölerek her birine sprint adını verirler. Projeyi böler ve her bir sprinti teker teker geliştirir. Ekip içi iletişim bu modelde çok önemlidir. Bu nedenle hergün SCRUM MEETİNGS toplantıları yapılır. Bu modelde 3 temel kavram vardır; roller, toplantılar ve araçlardır.

**Roller:** Ürün yöneticisi, Scrum yöneticisi ve Scrum takımından oluşur. Projenin beyni ürün sahibidir. Scrum yöneticisi, takımı kontrol eder. Takım 5-9 kişiden oluşan birbirleriyle iletişim içerisinde olan bir ekiptir.

**Toplantılar:** Hergün Scrum toplantıları yapılır toplantılarda geliştiricilerin önceki gün ne yaptıkları, karşılaştıkları sorunları ve bugün ne yapacakları hakkında konuşulur.

**Araçlar:** Ürün gereksinim dökümanı oluşturulur. Basitçe yapılması gerekenler yazılır. Sprint Dokümanı oluşturulur. Sprint dokümanı ürün gereksinim dokümanına takiben oluşturulur ve amacı her sprintin ona uygun ayarlanmasıdır ayrıca bu dokümanı sadece takımdakiler değiştirebilir. Sprint kalan zaman grafiği ise yapılan işin ne seviyede olduğu ve aslında planlanan zamana göre nasıl bir ilerlemede olduğunu belirtmek için hazırlanır. SCRUM günümzde en çok kullanılan geliştirme yöntemidir. Nedenleri ise;

\* Zamandan ve paradan büyük ölçekte tasarruf edilir.

\* Yüksek teknolojiler ve son gelişmelere kolaylıkla uyum sağlar.

\* Karmaşık ve gereksinimleri belli olmayan projeler için idealdir.

\* Ekip içi iletişimin fazlalığı ve hataların erken fark edilmesi.

\* Kullanıcıdan sürekli geri bildirim alması.

\* Diğer yazılım geliştirme metodlarıyla uyumlu olması.

\* Değişen gereksinime hızlı cevap verebilme.

\* Böl ve fethet uygulanmasıdır.

**Yazılım Geliştirme Modellerinin Karşılaştırılması**

Gelişigüzel ve barok modelleri yinelemeli olmadıkları ve dökümantasyonu işiniçinde tutmadıkları için günümüzde kullanılmıyan modellerdir.

Şelale modelinin kullanılması ve uygulanması basit, iş bölümü ve iş planlaması detaylıdır. Küçük ve özellikleri belli projelerde kullanılır.eskiden yaygın olarak kullanılıyordu ancak iletişim yetersizliğinden dolayı, büyük projelerde yetersiz olması ve değişimlere kapalı olması nedeniyle günümüzde çok kullanılmıyor.

V modeli şelale modelinin gelişmiş versiyonu diyebiliriz. Kullanımı ve takibi kolay olması, onaylama ve doğrulama işlemlerinin planın erken aşamasında uygulanabilir olması avantajlıdır ancak fazlar arası tekrarlamalara imkân tanımaması, risk çözümleme basamaklarının bulunmaması gibi sorunlar bulundurur.

Spiral modelde kullanıcılar sistemin içerisindeler. Risk analizi iyidir. Büyük projeler için idealdir. Hatalar erken giderilir. Ancak karışıktır ve çok üzün sürer.

Artımsal Geliştirme modelinde müşteri her seferinde ürünü gördüğü için işlevseldir proje daha güvenlidir. Ancak her artımı tamamlamak için sistemin detaylı olarak tamamlanması lazımdır ve kendi içlerinde tekrarlanmazlar.

Kodla ve Düzelt modeli küçük programlar içindir. Bakım safhası zordur. Büyük projeşer için çok maliyetlidir. Tek kişinin isteğine göre hazırlanır en sonuna kadar kodlanır ve teslim edilir. Dökümantasyon gibi aşamalar yoktur.

Evrimsel geliştirme metodunda kullanıcılar gereksinimleri daha iyi anlarlar, hatalar azdır ancak bakımı zordur.

Çevik modellerde müşteri ile iletişim yüksektir. Hatalar minimumdur. Alışılması basittir. Değişime açık ve esnektir. Ancak dökümantasyon çok detaylı değildir. İhtiyaçlar sık sık değiştiğinden çalışma saatleri fazla olabilir. Günümüzde çokça kullanılır.

**Hangi Projede Hangi Modeli Kullanmalıyız?**

Belirsizliğin az olduğu, tanımların belirgin olduğu projelerde v modelini kullanmak uygundur.

Büyük kitlelere ulaşacak projeşerde evrimsel geliştirme metodunu kullanmak uygundur.

Büyük maliyetli ve uzun süren projelerde spiral ve artımsal geliştirme modelini kullanmak uygundur.

Orta ve küçük, uzun sürmeyen projelerde çevik modelleri kullanmak uygundur.

Kişiye özel zaman sıkıntısı olmayan küçük projelerde kodla ve düzelt metodu kullanmak uygundur.

Küçük ve özellikleri iyi tanımlanmış projelerde şelale modelini kullanmak uygundur.

**Scrum Günümüzde Neden Popüler?**

Scrum ekibi ve müşteri iç içe oldukları için değişen istekleri çabucak yerine getirebilirler. Hergün yapılan scrum meetings toplantıları sayesinde ekip içi iletişim artar bu nedenle hatalar daha çabuk bulunur ve daha çabuk çözüm üretilir aynı zamanda erkenden hata bulunduğu için maliyeti de artmadan sorun çözülür. Sadelik, netlik, ve herşeyin planlı ve programlı olması ekiplerin scrum kullanmasını kolaylaştırır. Bu nedenle dünya üzerinde bu kadar popüler bir modeldir.

Yiğit Emir SANURA No: 200601071

Medium hesabındaki makalenin linki:

https://yigitesanura.medium.com/yazılım-yaşam-döngüsü-cc061918feec