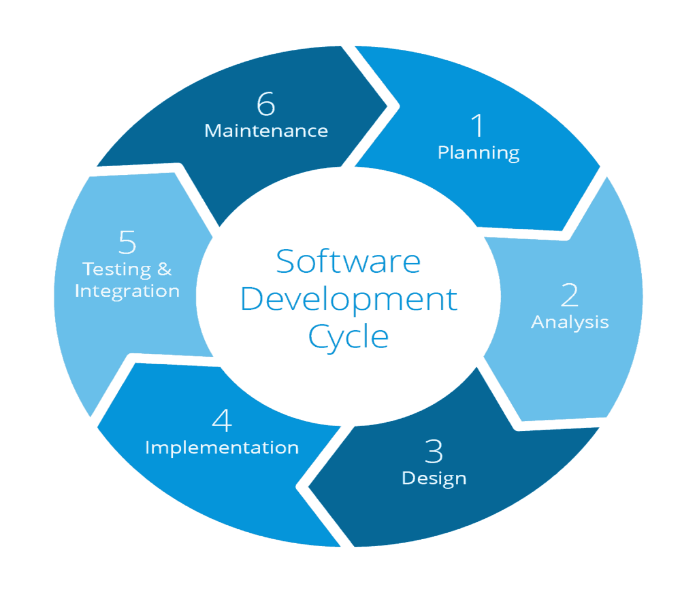
YAZILIM YAŞAM DÖNGÜSÜ

Yazılım yaşam döngüsü, bir yazılımın gerçekleşmesinden bitimine kadar devam eden aynı zamanda bakım işlemlerini de kapsayan bir süreçtir.6 aşamadan oluşmuştur sırasıyla şunlardır:

Planlama, Çözümleme, Tasarım, Gerçekleştirme, Test, Bakım

Planlama: Yazılımın planlandığı ve iş bölümünün yapıldığı aşamadır.

Analiz: Yazılımın hazırlanma süresi ve oluşabilecek risklerin ele alındığı aşamadır.

Tasarım: Yazılımın kaba taslak tasarısının yanı sıra ayrıntılı tasarısının da yapıldığı ve aynı zamanda nasıl devam edileceğinin de belirlendiği aşamadır.

Gerçekleştirme: Yazılımın kodlanma ve gerçekleşme aşamasıdır.

Test: Gerçekleştirme evresinde yazılan kodların test edildiği aşamadır.

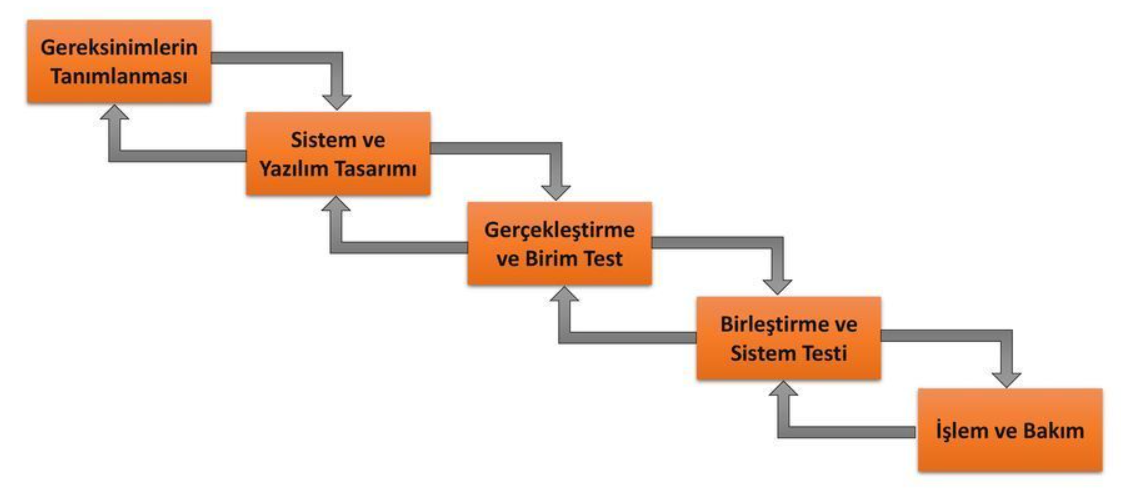
Bakım: Yazılım teslim edildikten sonra sistemdeki olası hataların düzeltildiği veya gerekli güncellemelerin yapıldığı aşamadır.

YAZILIM YAŞAM DÖNGÜ MODELLERİ

Yazılım yaşam döngüsünde birden çok model kullanılmaktadır. Sadece tek model olmama sebebi ise her projenin aynı yöntemle çözülemeyeceği gerçeğidir. Bu nedenle birden fazla model bulunuyor fakat bazıları modern sorunlara çözüm olamadığı ve karmaşıklığı sebebiyle günümüzde kullanılmamaktadır. Modellerin bazılarını tanıtayım.

Gelişigüzel Model: Herhangi bir yöntem içermez bu yüzden bakım kısmı oldukça zordur.1960’larda kullanılmıştır. Günümüzde kullanılmamaktadır.

Barok Model: Yaşam döngüsü adımları doğrusal bir şekilde ele alınmıştır fakat aşamalar arası geri dönüşler belirsizdir. Dokümantasyon işlemi ayrı bir süreç olarak ele alınmıştır.1970’lerde kullanılmıştır. Günümüzde kullanılmamaktadır.

Şelale Model: Geleneksel yazılım geliştirme modelidir. Eskiden en çok kullanılan yaşam döngü modelidir. Yazılım aşamaları en az birer kez tekrar edilir. 

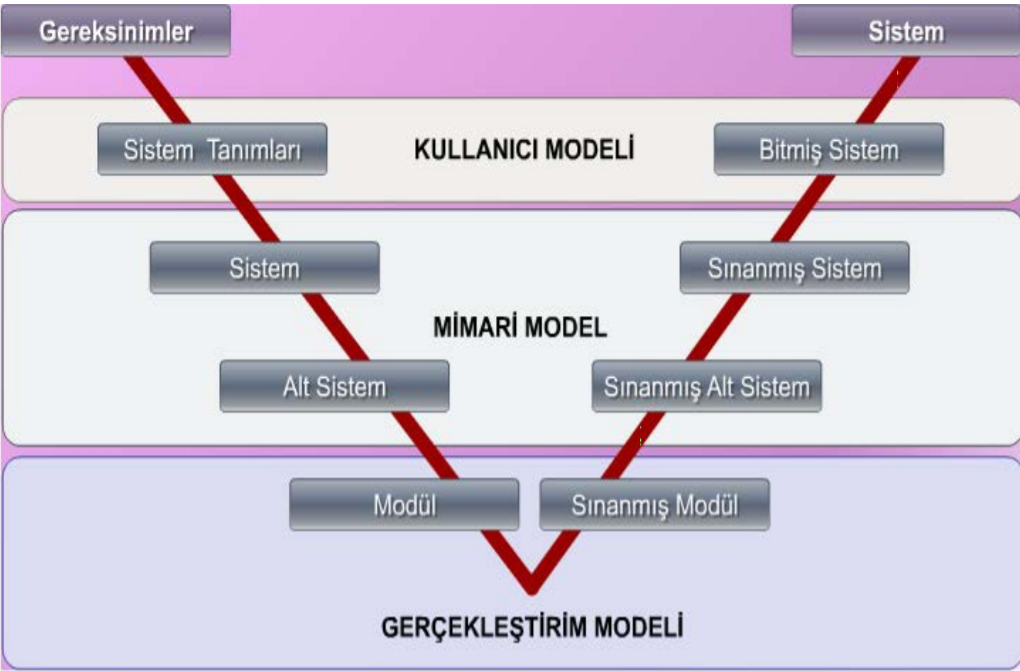
Bu modelde koddaki hatalar süreç bittikten sonra tespit edildiği için bu modeli küçük projelerde uygulamak daha uygun olur. İyi anlaşılmış ve kısa süreli projeler için uygun bir modeldir. Bir aşama bitmeden diğerine geçilmez. Şelale modelinin eksiklikleri şunlardır:

Projeler genelde uzun süreç gerektirir. Şelale modeli bu tarz projelerde kullanılamamaktadır.

Şelale modelinde kullanıcı sürecin içinde olmadığı için olası hatalar geç fark edilebilir bu da maliyeti artırmanın yanı sıra daha uzun teslim edilmeye sebep olur.

Yazılımcılar ağırlıklı olarak kodlama ve programı çalıştırma amaçları oldukları için bu modelde de kodlama kısmı küçük bir kısımda var olduğundan mutsuz olurlar.

Şelale modelini barok modeliyle kıyaslayacak olursak dokümantasyonu ayrı bir süreç değil de bir parçası olarak görüyor. Ayrıca aşamalar arası geri dönüşler tanımlıdır.

V Süreç Model: 

V süreç modeli, şelale modelinin uygulanmasını,“Üretim”ve”Deneme”işlemlerinin ne zaman yapılacağını vurgulayarak daha anlamlı hale gelmiştir.V modelinin sol tarafı üretim sağ tarafı ise deneme işlevlerini içerir.Bu model 3 parçadan oluşmaktadır.Kullanıcı Modeli,Mimari Model, Gerçekleştirim Model.

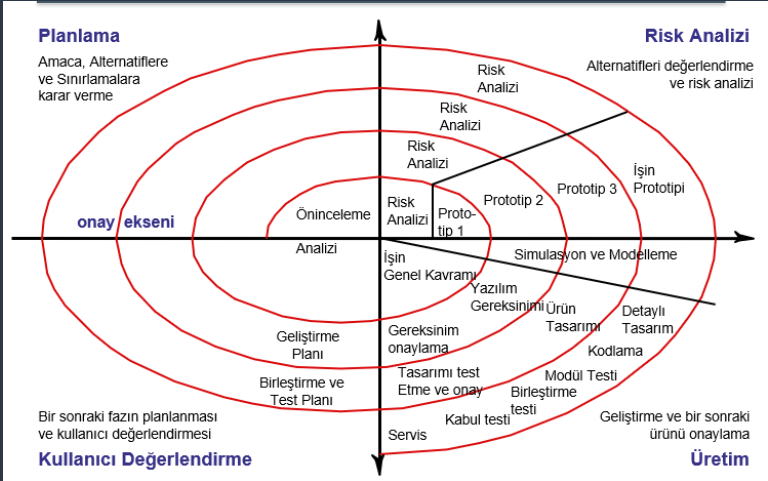
Kullanıcı Modeli:Geliştirme sürecinin kullancıyla olan ilişkisini tanımlamaktadır.

Mimari Model:Sistem tasarımı ve oluşacak alt sistem ve tüm sistemin deneme işlevlerini içerir.

Gerçekleştirim Model:Yazılımın kodlanmasını ve deneme işlevlerini içermektedir.

V Süreç Modelinin eksikleri şunlardır:Aşamalar arasında tekrar bulunmaz ve risk çözümleme için ayrı bir yer bulunmamaktadır.

Helezonik (Spiral) Model:



Helezonik modelde risk analizi her safhada olduğundan hatalar erken giderilebilir. Ayrıca prototip oluşumu her aşamada olduğundan dolayı kullanıcının her aşamada yazılımı görme fırsatına sahiptir.4 aşamadan oluşmuştur:

Planlama: Üretilecek ara ürün için planlama, bir önceki üretilen ara ürün ile bütünleştirme çalışması.

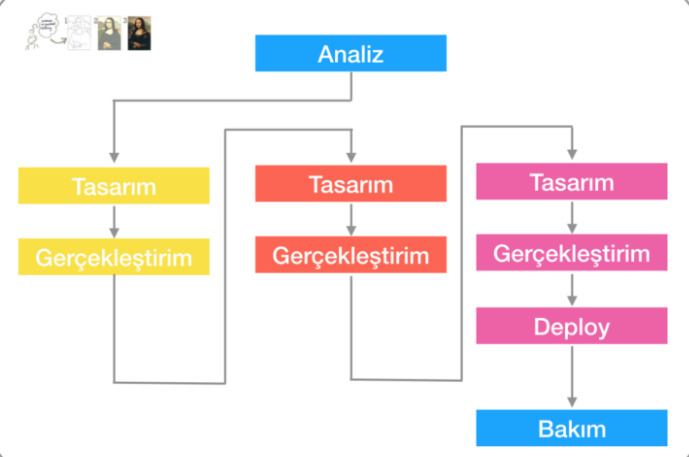
Risk Analizi: Risklerin tespit edilişi ve çözümü

Üretim: Ürünün veya ara ürünün gerçekleşmesi

Kullanıcı Değerlendirme: Ürün veya ara ürün hakkında kullanıcının denemeleri ve değerlendirmeleri

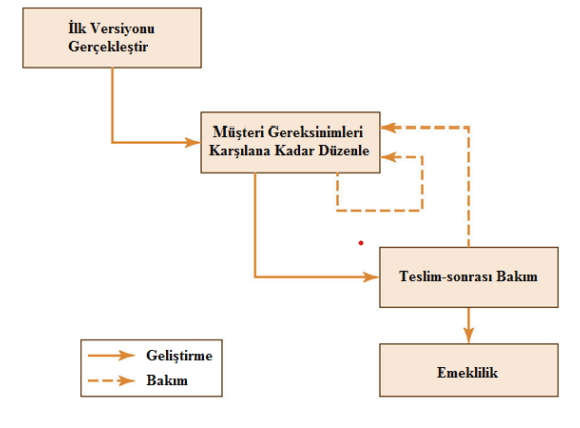
Helezonik Modelin eksikleri şunlardır: Küçük ve riski az olan projeler için pahalı olması, karmaşıklık içermesi ve fazla doküman içermesidir.

Artımlı Geliştirme Model:



Sistem parçalar halinde teslim edilir. Kullanıcı gereksinimleri ön planda tutulur ve gereksinimler önem sırasına göre sıralanıp teslim edilir. Parçaların her biri bir ara ürün olur ara ürünler kullanıcı tarafından kullanılır. Ara ürünler sürekli geliştirilerek üretilir. Üretim yaparken ürünü kullanılabilir halde oluyor bu modelde. Bu model uzun zaman alan ve tamamlanmamış halde çalışabilen türdeki yazılımlar için uygundur. Başarısız olma ihtimali düşüktür.

Artımlı Geliştirme Modelinin eksikleri şunlardır: Parçalar kendi içinde tekrar edemez. Bir parça bitmeden herhangi bir parçaya müdahele edilemez.

Kodla Düzelt Yaşam Döngü Model: 

Küçük programlar için uygundur, hemen ürün gerçekleşir, emeklilik safhası bulundurur. Dokümantasyon olmadığından dolayı bakım işlemi oldukça zordur. Ürün hazırlanır ve kullanıma sunulur.

Çevik Modeller: Projelerin başarı oranını artırmak, ekip içi iletişimi kuvvetlendirmek, maliyeti düşürmek, teslim süresini kısaltmak amacıyla bulunmuş modeller topluluğudur. Çevik modellerin bazıları şunlardır:

Extreme Programming (XP): Kent Beck 1999 yılında yazılımı geliştirmek için bulmuştur. Bu model 4 aşamadan oluşmaktadır: İletişim, Basitlik, Geri Bildirim, Cesaret.

İletişim: İletişim projelerde çok önemlidir. XP ise ekip iletişimine önem vererek projenin daha sağlıklı olmasını sağlıyor.

Basitlik: Günün ihtiyaçları doğrultusunda esnek ve basit bir sistemle çözer. Karmaşık çözümler XP e aykırıdır. Uzun dokümantasyon tercih edilmez.

Geri Bildirim: Müşterilerle yazılım ekibi ara ara buluşup bu noktadan sonra nasıl devam edileceğine karar verirler. Anlaşmazlıklar giderilir.

Cesaret: Kodu yanlış yazmaktan korkmamalı gerekirse en baştan tekrar yazabilecek olmalısınız. Yazılımdan korkmak teslim sürecini uzatır.

SCRUM: Jeff Sutjerland ve Ken Schawaber tarafından 90’ların ortasında geliştirilen bir yaklaşım. Her yere uygulanabilir. Büyük projeleri bölerek parçalara ayırır. Bu parçalara sprint adı verilir. Her sprint ayrı ayrı geliştirilir. İletişim o kadar önemli ki her gün “SCRUM MEETİNGS” toplantı yapılır. SCRUM 3 kavramdan oluşmaktadır; Roller, Toplantılar ve Araçlar.

Roller: Ürün sahibi, SCRUM yöneticisi ve SCRUM takımı olmak üzere 3 rol bulunuyor. Ürün sahibi, projenin yöneticisi gibidir. SCRUM yöneticisi

Toplantılar: SCRUM’ın vazgeçilmezlerindendir. Toplantı gününde önce neler yapıldığı, karşılaştıkları hatalar ve sonrasında neler yapılacağı belirlenir. Sprintler de ayrı ayrı gözden geçirilir.

Araçlar: 2 tür doküman vardır: Sprint ve Ürün gereksinim

Ürün gereksinim: Proje süresince gereken iş elemanlarının listesidir. Canlı olduğundan eleman eklenip veya çıkarılabilir. Bu liste kullanıcı hikayeleri baz alınarak oluşturulur.

Sprint: Sprint dokümanı ürün gereksinim dokümanına takiben oluşturulur ve amacı her sprintin ona uygun ayarlanmasıdır ayrıca bu dokümanı sadece takımdakiler değiştirebilir. Ayrıca sprint kalan zaman grafiği de yapılan işin zamana göre nasıl ilerlediğini hakkında bilgi verir.

HANGİ PROJEDE HANGİ MODELİ KULLANMALIYIZ?

İş tanımlamnın iyi olduğu bilgi teknolojileri projeleri için V Modeli gayet uygun bir modeldir.

Büyük, maliyetli ve uzun süren projelerde spiral model artımsal geliştirme modeli uygundur.

Orta veya küçük ebatlı ve kısa süren projelerde ise çevik modeller daha uygundur.

Kişisel projelerde ve zaman sıkıntısı olmayan projelerde kodla ve düzelt metodu daha uygundur.

Küçük ve iyi anlaşılmış projeler için ise çağlayan (şelale) modeli daha uygundur.

SCRUM GÜNÜMÜZDE NEDEN POPÜLER?

SCRUM en çok kullanılan yazılım geliştirme yöntemidir. Tek yönlü olmadığından sistem geliştirilmesinde de kullanılır. Bunlara ek olarak asıl sebepler şunlardır:

-Kısa zamanda düşük maliyet ile gerçekleşmesi

-Gelişime açık ve teknolojiye uyum sağlayabilir

-Yinelemeli olması

-Hızlı reaksiyon verebilmesi

-Böl ve fethet yaklaşımında bulunması

-Kullanıcıyla iç içe olması ve geri bildir gerektirmesi

-Ekip içi iletişimin üst seviyede olması

-Karmaşık gözüken ve tam anlaşılamamış projeler için ideal olması

EN İYİ YAZILIM GELİŞTİRME MODELİ HANGİSİ?

Gelişigüzel ve barok modelleri kullanılmamaktadır. Çünkü tekrar kısmı bulundurmuyorlar aynı zamanda dokümantasyon küçük bir rol aldığı için kullanılmamaktadır.

Şelale modeli kullanımı ve uygulanması kolay, iş bölümü ve iş planı detayladır. Küçük ve iyi anlaşılmış projelerde kullanılır. Artık günümüzde pek kullanılmamaktadır çünkü büyük projeler için yetersiz durumda kalıyor ve değişime açık değildir.

V Modeli şelale modelinin bir tık üstüdür ve daha çok bilgi teknoloji projeleri için uygundur. O yüzden kullanımı kolaydır. Ancak fazlar arası tekrar olmaması ve risk çözümleme basamaklarının olmaması büyük projelerde kullanılamamasının sebepleri arasındadır.

Spiral modelde ise kullanıcılar ile yazılımcılar iç içe olduğundan hatalar erken telafi edilir, risk analizi her aşamada bulunur, büyük projeler için uygundur. Teslim süresinin uzun ve karmaşık bir yapıya sahip olması ve dokümantasyon fazlalığından büyük projelerin vazgeçilmez modelidir.

Artımlı Geliştirme modelinde ise müşteri gereksinimi ön planda tutulduğundan daha güvenli bir proje haline gelir. Her parça teslim edilişinden sonra kullanıcıdan geri dönüş alınıyor bu da hata oranını düşürüyor. Ancak kullanıcının her dönütü detaylı incelenmesi gerektiğinden süre uzar ve parçalar kendi içinde tekrarlanamaz.

Kodla ve düzelt modeli küçük işler için idealdir. Ancak bakım safhası çok zordur ve gelişime açık değildir. Büyük projelerde kullanılamaz ve maliyeti çok yüksektir. Kişisel olduğundan dokümantasyon gibi olaylar bulunmaz sadece kodlanır ve teslim edilir.

Çevik modellerde hem müşteri ile hem de ekip içi iyi bir iletişim olduğundan hata olasılığı azdır. Değişime uygundur. Gereken ihtiyaçların sürekli değişmesi çalışma saatini uzatabilir. Dokümantasyon vardır fakat fazla önem verilmez. Son zamanlarda kullanımı baya yaygınlaşmıştır.

KAYNAKLAR

<https://www.techwell.com/2013/02/why-scrum-so-popular>

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

<https://www.quora.com/Why-is-the-Scrum-process-so-popular-in-the-software-industry>

<https://enprobilisim.com/yazilim-gelistirme-sureci-modelleri-sdmp/>

<https://medium.com/@secilcor/scrum-nedi%CC%87r-6a4326951dd8>

<https://medium.com/@omerharuncetin/yaz%C4%B1l%C4%B1m-ya%C5%9Fam-d%C3%B6ng%C3%BC-modelleri-543c7879a742>

<https://caglartelef.com/yazilim-yasam-dongusu/>

-Doç. Dr. Deniz Kılınç, Bakırçay Üniversitesi Yazılım Mühendisliğine Giriş Dersi 2. ve 3. Hafta Sunumları