ÖMER KARAKEÇE 220601001

İzmir Bakırçay Üniversitesi Bilgisayar Mühendisliği 2. Sınıf

Linkedin : https://www.linkedin.com/in/%C3%B6mer-karake%C3%A7e-5774011a9/

Github : https://github.com/omerkrkcc

Medium : https://medium.com/@omerrkrkc

Yazılım Yaşam Döngü Modelleri

Yazılım Yaşam Döngüsü Nedir ? Neden İhtiyaç Duyarız ?

Yazılım süreç gerektiren bir yapıdır. Bu yapıyı kurmak, modernize etmek, çıkabilecek sorunlara karşı temkinli olmak yani kısacası yazılım geliştirme sürecinin yönetilmesi ve kontrol edilmesi bir ihtiyaçtır. 1960'lı yıllarda yazılımın gelişmeye başlamasıyla birlikte yeni sorunlar ve zorluklar ortaya çıkmıştır. Yazılım ile ilgilenen kişiler bu sorunların üstesinden gelmek için bir arayış içerisine girdiler. Ve böylelikle yazılım yaşam döngüleri ortaya çıktı. Yazılım yaşam döngüleri ile yazılımın kalitesi, verimliliği artarken, projenin bütçesi ve zaman yönetimi de doğru bir şekilde planlanıp yönetilir.

Yazılım yaşam döngüleri 5 temel aşamadan oluşur.

• Planlama : Yazılım projelerinin geliştirilmesi ve yönetilmesi sürecinde tüm aşamaların kontrollü bir şekilde gerçekleşmesi için oluşturulan bir çatıdır.

• Analiz : Yazılımda ihtiyaç duyulan işlevselliğin ve sistem gereksinimlerinin detaylı olarak incelendiği aşamadır.

• Tasarım : Elde edilen gereksinimler ile bu gereksinimlere cevap oluşturacak bir yazılımın nasıl çalışacağı tasarlanır.

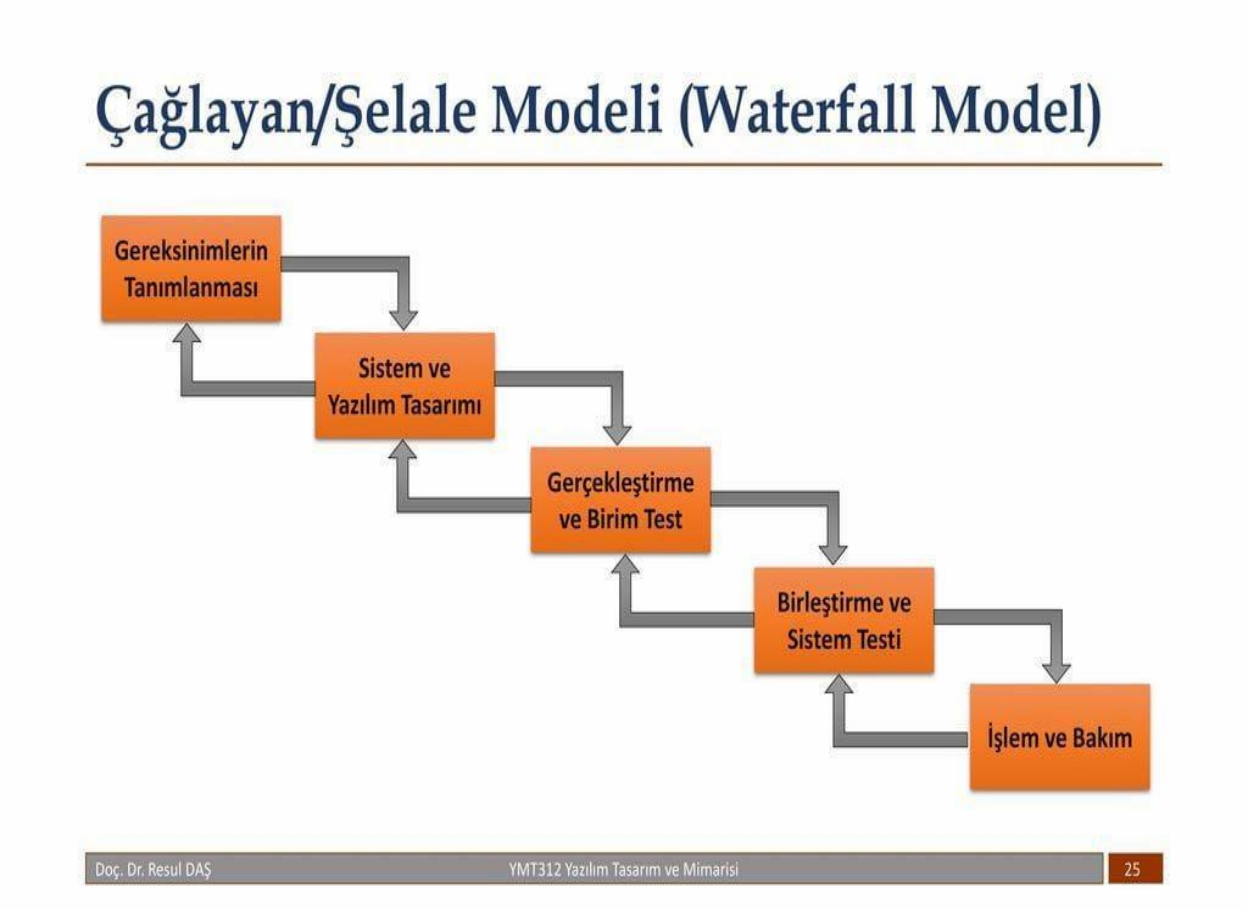
• Gerçekleştirim (Kodlama ve Test) : Projenin kodlama ve test aşamasının başladığı kısımdır.

• Teslim ve Bakım : Ürünün ortaya çıkması ile birlikte tesliminin gerçekleştiği aşamadır. Teslim aşaması ile birlikte bakım aşaması da başlar. Ürünün tesliminden sonra güncelleme ve bakım sürecinin bulunduğu aşamadır.

Yazılım işlevleri ve ihtiyaçları sürekli geliştiği ve değiştiği için, yazılım yaşam döngüsü artık doğrusal bir süreç olarak düşünülemez. Bunun yerine, yazılım yaşam döngüsü artık bir döngü şeklinde düşünülmelidir. Bu sayede sürekli gelişmeleri ve değişimleri göz önünde bulundurarak daha etkili bir şekilde yönetilebilir.

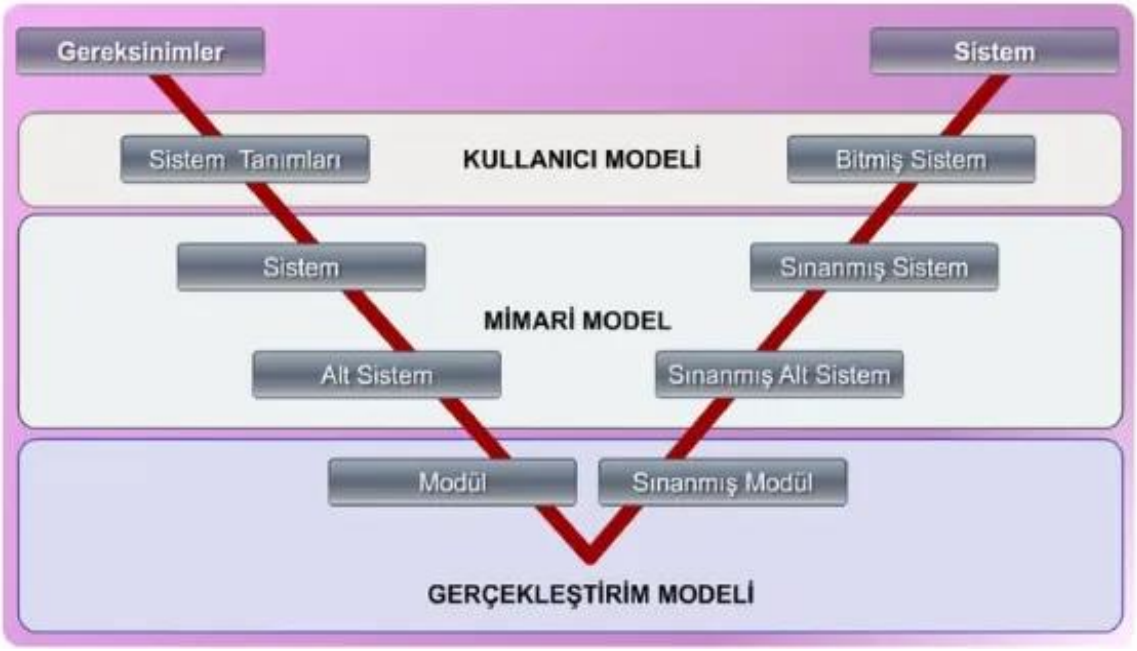
Yazılım Yaşam Döngü Modelleri

• Çağlayan – Şelale (Waterfall) Modeli



Şelale modeli bilinen en eski modellerden birisidir. Yazılım geliştirme sürecinin en sıradanlaşmış ve eski yaklaşımlarından biri olduğu için geleneksel model olarak da adlandırılır. Şelale modeli eskiden çok popüler bir yöntemken günümüzde değerini yitirmeye başlamıştır. Üretimi az zaman gerektiren, iyi planlanmış ve çok iyi tanımlanmış projelerde tercih edilir. Bunun sebebi aslında bu modelin esneklik, müşteri talebi ve ekip işbirliği konularında eksikliğidir. Çağlayan modelde belgeleme temel yapı taşıdır. Her aşamada belgeleme önemlidir ve başından sonuna kadar belgeleme yapılır. Çağlayan modelde aşamalar arası geri dönüşler tanımlıdır. Bu modelde yazılım geliştirme süreci lineer olarak ilerler. Yazılım, aşamaların en az birer kez tekrarlanması ile gerçekleşir. Çağlayan modeli statik (değişimlere elverişsiz) bir modeldir. Büyük projelerde sürekli yeni farklı ihtiyaç ve fikir çıktığından dolayı kullanımı tercih edilmez. Bu modelin en büyük sorunlarından biri de yazılım ekiplerinin hızlı ürün çıkartma ve sonucu görme eğiliminde oldukları için süreçin uzunluğu ekibi kod yazma dışındaki süreçlerde verimsiz kılar. İyi tanımlanmamış ya da büyük bir projede maliyet ve zaman yönetimi açısından kayıplara neden olur.

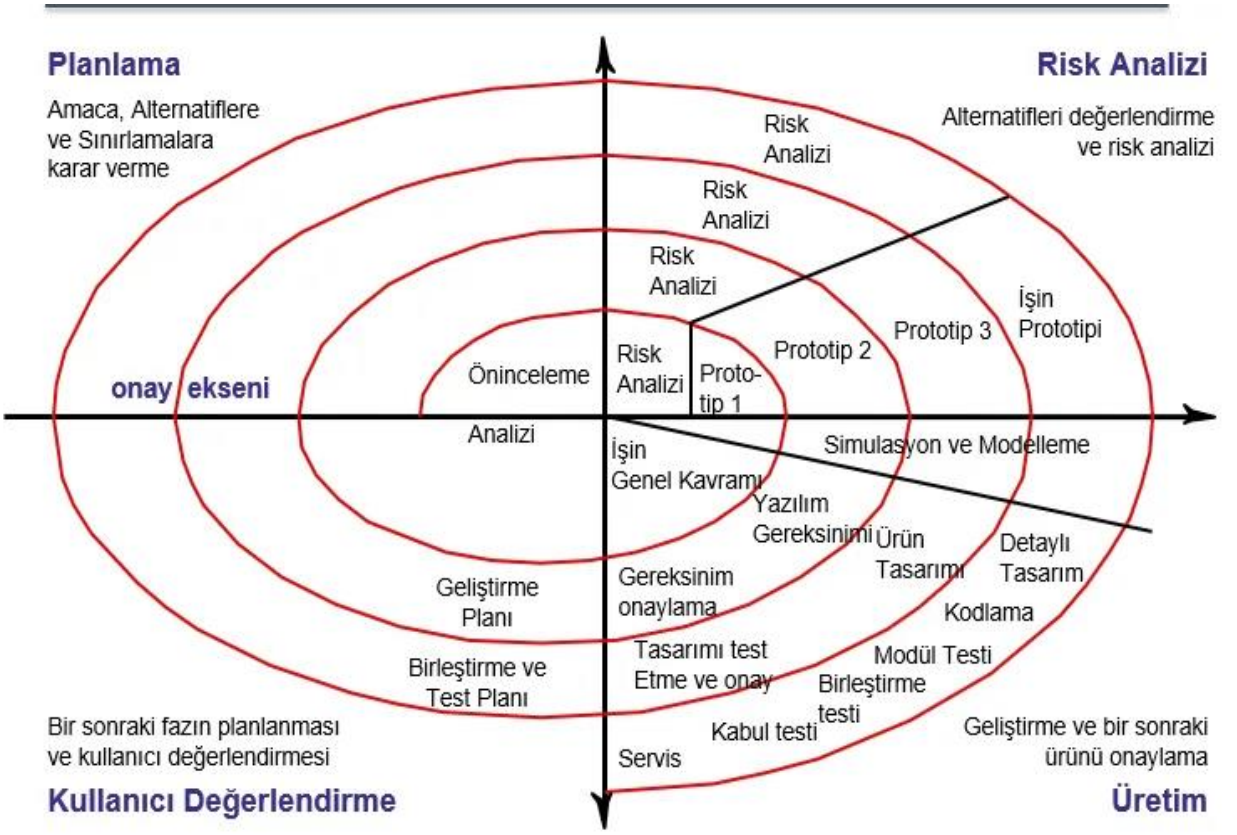
• **V Modeli**



V model şelale modelinin biraz gelişmiş halidir. Bu aşamada şelale modeline göre daha fazla test etme kısmına önem verilmiştir. Bu modelin sol tarafı üretim, sağ tarafı sınama (test) işlemlerinden oluştuğu için modelin yapısı V harfine benzer olduğundan dolayı V Model olarak adlandırılmıştır. Bu model kullanıcı modeli, mimari model ve gerçekleştirim modeli olmak üzere 3 temel aşamadan oluşur. Üst ve alt seviye tasarımı vardır. Üst seviyeler daha bütünsel bir tanımlama içeririr. Alt seviyelerde ise detaylı bir tasarım mevcuttur. Bu model belirsizliklerin az, iş tanımlarının belirgin olduğu projeler için uyumludur. Bu modelde proje yönetimi kolaydır. Ancak fazlar arasında tekrarlama ve risk çözümleme aktiviteleri yoktur. Bunlar bu modelin dezevantajı olarak değerlendirilebilir.

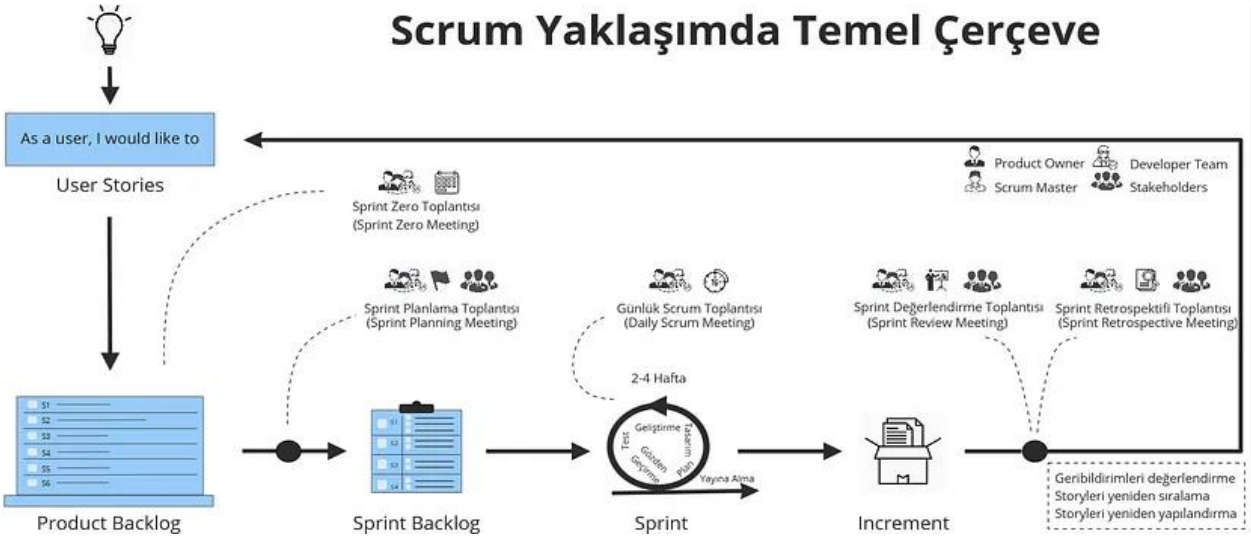
• **Spiral (Helezonik) Model**

Spiral model bir spiral etrafında spiral şeklinde dönmesi ve her döngünün bir fazı ifade etmesinden adını almıştır. Diğer modellerin yanı sıra risk analizi ve prototip yaklaşımı öne çıkar. Yinelemeli arttırımsal yaklaşım benimsenerek her bir spiralin riskleri ayrı ayrı ele alınır. Çağdaş ve popüler modellere çok yakın modellerden birisidir. Bu model planlama, risk analizi, üretim ve kullanıcı değerlendirmesi olmak üzere 4 temel aşamadan oluşur. Modeldeki prototip yaklaşımı ile kullanıcılar sistemi erken görebilirler. Geliştirmeyi parçalara bölüp en riski olanı çözümlemeye odaklanır. Bu model hataların erken fark edilmesine ve kodlamanın erken başlamasına, sınanmasına olanak tanır. Karmaşık yapısı spirali sonsuzluğa sürükleyebilir. Ara adımlarının çokluğu sebebiyle çok fazla dökümantasyona ihtiyaç duyulur. Spiral model küçük ve düşük riskli projeler için uygun değildir. Spiral modelin kullanımı daha çok karmaşık ve büyük ölçekli projelerde gerçekleştirilir. Savunma sanayi, uzay ve haberleşme projeleri, siber güvenlik gibi yüksek riskli yazılım ihtiyacının bulunduğu alanlarda spiral model kullanımı tercih sebebidir.



- **Scrum Modeli**

Scrum bir çevik yazılım geliştirme yaklaşımıdır. Scrum modelini sadece yazılım geliştirme yaklaşımı olarak düşünmek yanlıştır. Scrum yazılım geliştirme yaklaşımının yanı sıra bir proje yönetim yaklaşımıdır. Böylelikle yazılım geliştirme projeleriyle sınırlı kalmayıp her projeye uygulanabilir bir yapıya sahiptir. Scrum belirsiz ve karmaşık projelerde en çok kullanımı tercih edilen modellerden biridir. Scrum belirsiz ve karmaşık projelerde adım adım yazılmı geliştiren ekipler için uygundur. Scrum çalıştığı projelerdeki kompleksliği 3 ilke ile azaltır. Bunlar şeffaflık, denetleme ve uyarlamadır. Şeffaflık projenin işleyiş yapısına ışık tutar. Proje ekibinin hataları hızlı fark edebilmelerine ve bunlarla ilgili aksiyon almalarına olanak tanır. Bir iterasyonun 30 günü aşmaz. Scrum müşteri memnuniyeti, çalışan mutluluğuna ve ekip içi iletişime çok önem verir. Günlük kısa toplantılarla (Scrum Daily Meeting) sürekli iş takibi yapılır.



Scrum yapısında müşteri tarafından tanımlanan istekler iki ve dört haftalık ‘Sprint’ adı verilen dönemde belirli bir işin tamamlandığı ve incelemeye hazır hale geldiği bir dönemdir.

Sprint plan, tasarım, geliştirme, test, gözden geçirme ve yayına alma kısımlarını içeren çok aşamaları bir bütündür.

Scrum 3 temel kavramdan oluşur. Bunlar Roller, Toplantılar ve Araçlardır.

• **Roller :** Scrum takımı genellikle 5 ile 10 üyeden oluşur. Eğer projeniz büyükse ayrı scrum takımlarına bölünür. Ve bu takımlar ortak bir projede çalıştıkları için sürekli iletişim ve koordinasyon halinde olmalıdırlar. Büyük projelerde kullanılan scrum yapılarına Scrum of Scrums (SoS) denir. Bir scrum takımı Ürün Sahibi (Product Owner), Scrum Yöneticisi (Scrum Master) ve Scrum Takımından (Scrum Team) oluşur. Karıştırılmamalıdır ki bir Scrum yapısı Proje Yöneticisi içermez.

Product Owner : Scrum sürecini Product Owner başlatır. Product Owner müşteri tarafından görevlendirilmiştir. Ürünü temsil eder. Ürün ile müşteri arasındaki iletişimi, sürecin başarılı bir şekilde sürdürülmesini ve ürünün müşteriye teslim edilmesinden sorumludurlar.

Scrum Master : Product Owner ile iletişim kuran kişidir. Takımın huzurlu ve motivasyonlu bir şekilde yönetiminden sorumlu kişidir. Takım içi sorunları çözümleyen takımın scrum değerlerini ve kurallarını takip etmelerine destekte bulunan kişidir. Takımın scrum ilkelerine sadık kalmalarına yardımcı olur.

Scrum Team : Ürünün oluşturulmasında etkili olan ekiptir. Genellikle kodlama işlerini yaptıklarından dolayı development team olarak da adlandırılabilirler. Fakat bu ekibin arasında veri bilimciler, tasarımcılar ve yazılım test uzmanlarıda bulunabilirler. Genellikle 5-10 kişiden oluşurlar.

• **Toplantılar :** Sprint Planlama (Sprint Planning), Günlük Scrum Toplantısı (Daily Scrum Meeting), Sprint Gözden Geçirme (Sprint Review) olmak üzere 3 aşamadan oluşur.

Sprint Planlama (Sprint Planning) : Takımın nasıl çalışacağı, takımın belirlenmesi, sprintlerde (sprint boyunca) neler yapılacağı, projenin nasıl ilerlemesi gerektiği, müşteri memnuniyetini sağlamak için en optimal gereksinim listesi, risk analizi ve malyiet hesaplama tahminlerinin bulunduğu kısımdır. Bu aşama 4 haftalık sprint için 8 saat, 2 haftalık sprint için 4 saat ile sınırlıdır.

Günlük Scrum Toplantısı (Daily Scrum Meeting) : Günlük Stand – Up olarakda isimlendirilen bu toplantı genellikle sabahları gerçekleştirilen ilerlemenin hızlı ve şeffaf şekilde ele alındığı takımın ilerleyişi, bu toplanın öncesindeki toplantıdan sonra yapılan çalışmalar, projeyle ilgili yaşanan problemlere çözüm üretme, gün içerisinde neler yapılacağı ve nasıl yapılması gerektiği hakkında bilgi edinilmesi gibi konuları içeren 15 dakika boyunca süren fikirlerin belirtildiği bir toplantıdır. Toplantı boyunca üyeler toplantıyı ayakta sürdürür ve bu toplantılar sprint boyunca gerçekleştirilir.

Sprint Review : Sprint review toplantısında sprint boyunca yapılan işler sunulur ve hangi işlerin tamamlandığı açıklanır. Yapılanlar incelenir ve kusurların giderilmesine yönelik çalışmalar yapılır. Ürün Sahibi kriterlere göre çalışmayı kontrol edip kabul ya da red eder.

• **Araçlar :** Ürün Gereksinim Dökümanı (Product Backlog), Sprint Dökümanı (Sprint Backlog), Sprint Kalan Zaman Grafiği (Burndown Chart) olmak üzere 3 temel aşamadan oluşur.

Product Backlog (Ürün Gereksinim Dökümanı) : Bu süreç Product Owner tarafından yönetilir. Product Owner müşteri ile iletişime girer ve müşteriden istenilen gereksinimler alınır. Bu gereksinimler öncelik sıralarına göre değerlendirilir. Ürün gereksinim dökümanı canlı bir dökümandır, yeni gereksinimler eklenip çıkartılabilir yani devamlı bir bakım ihtiyacı söz konusudur. Ürün gereksinim dökümanı kullanıcı hikayelerinden (user story) oluşur. Proje boyunca sürekli değişim ve güncelleme gerçekleştirilebilir.

Sprint Backlog (Sprint Dökümanı) : Product Backlog ile elde edilen iş ve görevlerin mevcut sprint üzerine aktarılmasıdır. Kısacası bir sprint için gerçekleştirilmesi gereken işlerin planlandığı aşamadır.

Sprint Kalan Zaman Grafiği (Burndown Chart) : Yatay eksende sprint günlerini, dikey eksende sprint boyunca kalan işi gösteren yapıdır. Kalan iş ve zaman arasında çizilen bir çizgi grafiğidir. İşin doğru ve zamanında yetişebilmesi için gereken sürenin yeterliliğini, ekibin verimli ve bir sonraki projelerde aktif bir şekilde rol alıp alamayacaklarını, işin ilerleyişinde sorun yaşanıp yaşanmayacağını, işin zamanında tesliminin analizinin yapıldığı kısımdır. Bu kısım scrum temel işlevlerinden şeffaflığı yansıtır.

Scrum modeli günümüzde en popüler ve yaygın kullanılan modellerden birisidir. Scrum modeli çevik yazılım geliştirme sürecini benimser. Kısa vadeli planlar ve küçük parçalar halinde yazılımın gelişmesini sağlar. Scrum ekipleri proje için gerekli olan minimum düzeyde döküman tutarlar.

Modellerin Karşılaştırılması

• Çağlayan – Şelale (Waterfall) Modeli

Şelale modeli basit, küçük, iyi tanımlanmış ve düşük riskli projelerde tercih edilir. Diğer modellere göre en kalıp ve eski modellerden birisidir. Bu model esneklik, müşteri talebi ve ekip işbirliği konularında diğer modellere göre eksiktir. Barok modelinde işlem sonunda dökümantasyon işlemleri test aşamasının sonunda yapılırken çağlayan modelde projenin başından sonuna dökümantasyon işlemi yapılır. Çağlayan modelde aşamalar arası geri dönüşler tanımlıdır. Barok modelinde ise aşamalar arası geri dönüşler belirsizdir. Yazılım, aşamaların en az birer kez tekrarlanması ile gerçekleştirilir. Çağlayan modeli statik (değişimlere elverişsiz) bir modeldir. Yukarıda bahsettiğim bazı olumsuz özelliklerinden dolayı günümüzde kullanımı çok azalmıştır.

• V Modeli

Şelale modelinin biraz gelişmiş halidir. Bu model belirsizliklerin az, iş tanımlarının belirgin olduğu projeler için uyumludur. Bu modelde proje yönetimi kolaydır. Kullanımı basittir. Ancak fazlar arasında tekrarlama ve risk çözümleme aktiviteleri yoktur. Bunlar bu modelin dezevantajları olarak değerlendirilebilir.

• Spiral (Helezonik) Model

Diğer modellerin yanı sıra risk analizi ve prototip yaklaşımı öne çıkar. Yinelemeli arttırımsal yaklaşım benimsenerek her bir spiralin riskini ayrı ayrı ele alır. Diğer modellere göre çağdaş ve popüler modellere çok yakın modellerden birisidir. Diğer modellere göre çok daha karmaşık yapıya sahiptir. Bu yüzden büyük ölçekli ve yüksek riskli projelerde tercih edilir. Karmaşık yapısı spirali sonsuzluğa sürükleyebilir. Ara adımlarının çokluğu sebebiyle çok fazla dökümantasyona ihtiyaç duyulur. Bu yapıda planlamaya çok önem verilir. Hatalar olabildiğince erken fark edilir ve aksiyon alınır. Spiral model genellikle uzun süreli projeler için tercih edilir.

• Scrum Modeli

Scrum yazılım geliştirme yaklaşımının yanı sıra bir proje yönetimi yaklaşımı olması ile diğer modellere göre farklıdır. Scrum belirsiz ve karmaşık projelerde en çok kullanılması tercih edilen modellerden birisidir. Scrum belirsiz ve karmaşık projelerde adım adım yazılım geliştiren ekipler için uygundur. Scrum müşteri memnuniyeti, çalışan mutluluğuna ve ekip içi iletişime çok önem verir. Günlük kısa toplantılarla (Scrum Daily Meeting) sürekli iş takibi yapılır. Scrum modeli bir çevik (agile) yazılım geliştirme modeli olduğu için süreci sürekli bir takip içerisinde tutmasından dolayı hatalar kolay fark edilir. Değişime ve yeniliğe açıktır. Basitliğe önem verir. Kısa vadeli planlar ve küçük parçalar halinde yazılımın gelişmesini sağlar. Scrum ekipleri proje için gerekli olan minimum düzeyde döküman tutarlar.

Hangi Projede Hangi Modeli Kullanmalıyız ?

Şelale modeli basit, küçük, iyi tanımlanmış ve düşük riskli projelerde tercih edilir. Eğer projeniz dökümantasyon işlemine önem veriyorsa çağlayan modeli iyi bir seçim olabilir. Çünkü şelale modelinde baştan sona her aşamada dökümantasyon işlemi bulunmaktadır. Şelale modelinde aşamalar arası geri dönüşler tanımlıdır. Yazılım, aşamaların en az birer kez tekrarlanması ile gerçekleştirilir. Esneklik, müşteri talebi ve ekip işbirliği konularının yanı sıra statik bir model olduğundan dolayı günümüzde kullanımı oldukça azalmıştır. Şelale modeli günümüzde yüz tanıma ve nesne algılama gibi bilgisayar görüsü (computer vision) alanlarında kullanılır.

V Modeli belirsizliklerin az, iş tanımlarının belirgin olduğu projeler için uygundur. Proje yönetiminin kolaylığı kullanım basitliği gibi sebeplerden dolayı tercih sebebidir. Ancak fazlar arası tekrarlamanın olmaması, risk çözümleme aktivititelerinin bulunmaması bu modelin dezevantajlarındandır. V modeli günümzde medikal cihaz yazılmıı geliştirme süreçlerinde kullanılır.

Spiral model risk analizinin ve prototip yaklaşımının önemli olduğu projelerde ön plana çıkar.Diğer modellere göre daha yeni bir modeldir. Karmaşık bir yapıya sahiptir. Büyük ölçekli ve yüksek riskli projeler için tercih edilir. Ara adımların sıklığından dolayı çok fazla dökümantasyona ihtiyaç duyulur. Planlamaya oldukça önem verilip hatalar erken fark edilir. Spiral model genellikle uzun soluklu projeler için tercih edilir. Spiral model günümüzde savunma sanayi, uzay ve haberleşme projeleri, siber güvenlik, finans uygulamaları, tıp gibi yüksek riskli projelerde tercih edilir.

Scrum modeli belirsiz ve karmaşık projelerde kullanımı en çok tercih edilen modellerden birisidir. Scrum belirsiz ve karmaşık projelerde adım adım yazılım geliştiren ekipler için uygundur. Eğer müşteri memnuniyeti, ekip içi mutluluk ve ekip içi iletişime önerm veriliyorsa scrum model kullanılır. Bu model sürecin planlı ve doğru ilerleyişine odaklıdır. Böylelikle hatalar çabuk fark edilir. Bu modelde ilgili dökümantasyonlar gerekli durumlarda hazırlanır. Kısa vadeli planlar ve küçük parçalar halinde uygulanır. Scrum modelinde küçük, büyük proje diye ayrım yapılmaz. Scrum modeli hemen hemen her türlü projeye uygulanabilecek esnek bir yapıya sahiptir. Ancak yapısı gereği büyük ve karmaşık projelerde ön plana çıkar.

Barok modelinde dökümantasyon test işleminden sonra yapılır. Aşamalar arası geri dönüş belirsizdir. Bu gibi sebeplerden dolayı günümüzde nerdeyse tercih edilmeyen modellerden birisidir.

Gelişigüzel modelde belirlenmiş bir model ya da yöntem bulunmaz. Genellikle kişiye bağlı yazılım geliştirme şeklinde yapılır. Esnek değildir. Değişime uygun bir model değildir. Günümüz projelerinin karmaşıklığı ve büyüklüğü karşısında yüksek maaliyetlere sebep olabilir. Bu sebepten dolayı günümüzde kullanımı yok denecek kadar azdır.

Evrimsel model pilot uygulama kullan, test et, güncelle ve diğer birimlere taşı çalışma mantığında ilerler. Evrimsel model günümüzde biyoloji, mühendislik, finans, yapay zeka, sanat ve tasarım gibi alanlarda kullanılır.

Kodla ve düzelt dökümantasyon gerektirmeyen çok küçük, düşük risk içeren genellikle bireysel projelerde kullanılan bir yöntemdir. Büyük projelerde uygulanması halinde maliyet çok fazla büyür. Hesap makinesi gibi birkaç yüz satırdan oluşan programlarda kodla ve düzelt kullanılabilir.

Artırımsal geliştirme modeli gereksinimlerin önemine göre teslim edilecek artırımları belirler. Divide and Conquer (Böl ve Yönet) yaklaşımını benimser. E ticaret platformları, sağlık yazılımı, mobil uygulamalar, oyun geliştirme gibi alanlarda kullanılır.

Scrum Günümüzde Neden Popüler ?

Scrum yazılım geliştirme yaklaşımının yanı sıra bir proje yönetimi yaklaşımı olması ile ön plana çıkmıştır. Scrum modelinin kullanım yelpazesi geniştir. Scrum modelini büyük,küçük proje diye sınırlamak doğru değildir. Hemen hemen her türlü projeye uygulanabilecek esnek bir yapıya sahiptir. Ancak yapısı gereği büyük , belirsiz ve karmaşık modellerde kullanımı en çok tercih edilen modellerden birisidir. Bu model karmaşık ve belirsiz projelerde adım adım yazılım geliştiren ekipler için en uygun modellerden birisidir. Bir iterasyon 30 günü aşmaz. Bu modelin popüler olmasındaki en büyük etkenlerden birisi müşteri memnuniyeti, çalışan mutluluğu ve ekip içi iletişime çok önem vermesidir. Müşteri merkeziyette tutulur. Müşteri memnuniyeti için özen gösterilir. Günlük kısa toplantılarla (Scrum Daily Meeting) sürekli iş takibi yapılır. Scrum modeli çevik (agile) bir yazılım geliştirme modeli olduğu için süreci sürekli bir takip içerisinde tutar . Bu takip sayesinde hatalar kolay fark edilir. Daha çok kompleks projelerde kullanıldığından kompleksliği azaltmak için 3 temel ilkeye sahiptir. Bunlar şeffaflık, denetleme ve uyarlamadır. Şeffaflık projenin ilerleyişine ışık tutar. Scrum projelerinin tercihini başarı oranı etkiler. Scrum modeli ile gerçekleştiren projelerin başarı oranı %80'nin üzerindedir. Scrum projeleri kısa vadeli planlar ve küçük parçalar halinde yazılımın gelişmesini sağlarlar. Projeler esneklik, basitlik, sürekli değişim ve gelişim yapısını benimser. Scrum ekipleri proje için gerekli olan minimum düzeyde dökümantasyon tutarlar. Yukarıda bahsettiğim sebepler scrum modelinin günümüzün popüler ve en çok kullanılan modellerinden birisi olmasını sağlamıştır.

KAYNAKÇA :

- https://www.quora.com/Why-is-the-Scrum-process-so-popular-in-the-software-industry
- https://www.youtube.com/watch?v=9TycLR0TqFA
- 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://slideplayer.biz.tr/slide/12386328/%20
- https://www.agileturkey.org/
- **https://acikerisim.sakarya.edu.tr/bitstream/handle/20.500.12619/80049/T05405.pdf?sequence=1&isAllowed=y**
- **Doç. Öğr. Üyesi Zekeriya Anıl GÜVEN, İzmir Bakırçay Üniversitesi Yazılım Mühendisliği Temelleri Dersi 1. , 2. ve 3. Hafta Sunumları**