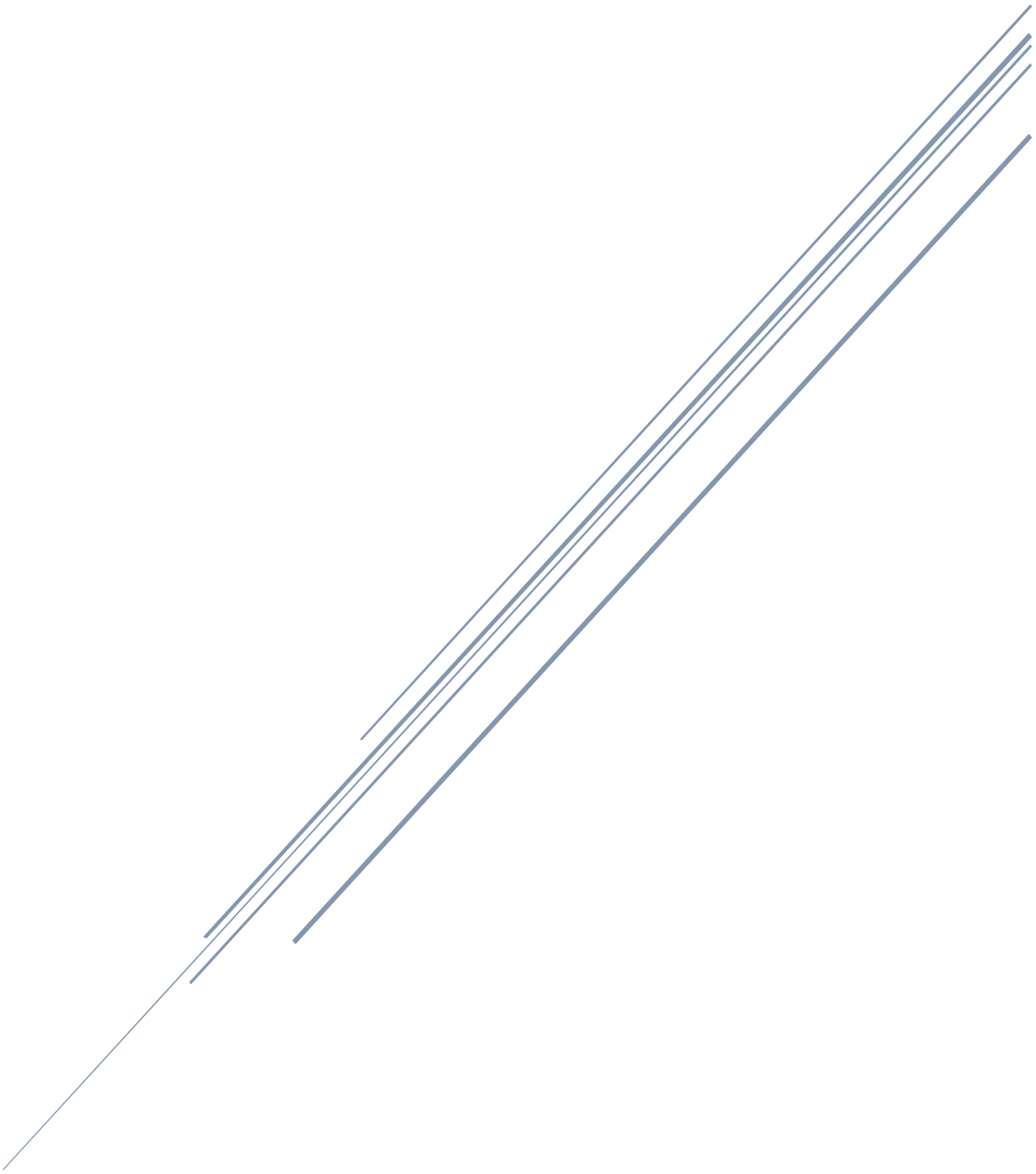
YAZILIM MÜHENDİSLİĞİNİN TEMELLERİ

YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ



Sidal Deniz BİNGÖL
210601009

YAZILIM YAŞAM DÖNGÜSÜ NEDİR?

Yazılım yaşam döngüsü, herhangi bir yazılımın üretim ve kullanım aşaması birlikte olmak üzere geçirdiği tüm aşamalardır. Yazılımla ilgili gereksinimler sürekli olarak değiştiği ve genişlediği için bu aşamalar bir döngü biçiminde ele alınır. Döngü içerisinde herhangi bir aşamada geriye dönmek ve tekrar ilerlemek söz konusudur. Yazılım yaşam döngüsünün temel adımları sırasıyla; planlama, çözümleme, tasarım, gerçekleştirme ve bakımdır.

Planlama

Üretilecek olan yazılımla ilgili olarak, personel ve donanım ihtiyaçlarının belirlendiği, fizibilite çalışmasının yapılıp proje planının oluşturulduğu aşamadır.

Çözümleme

Yazılım gereksinimlerinin ayrıntılı olarak çıkarıldığı aşamadır. Bu aşamadaki temel amaç, mevcut yapıdaki işlerin ortaya çıkarılması ve doğru olarak algılanıp algılanmadığının belirlenmesidir. UML diyagramlarının çizimlerine de bu aşamada başlanır.

Tasarım

Belirlenmiş gereksinimlere karşılık verecek yazılım ya da bilgi sisteminin temel yapısını oluşturma aşamasıdır. Mantıksal ve fiziksel tasarım olmak üzere ikiye ayrılır. Mantıksal tasarımda mevcut sistemin değil, önerilen sistemin yapısı anlatılır. Fiziksel tasarımda ise yazılımı içeren bileşenler ve bunların ayrıntıları incelenir.

Gerçekleştirme

Planlama ve tasarımı tamamlanmış projenin kodlandığı, test edildiği ve kurulumunun yapıldığı aşamadır.

Bakım

Yazılımın hatalarının giderildiği ve yazılıma ek yeniliklerin getirildiği aşamadır. Bu aşama yazılımın yaşam süresi boyunca devam eder.

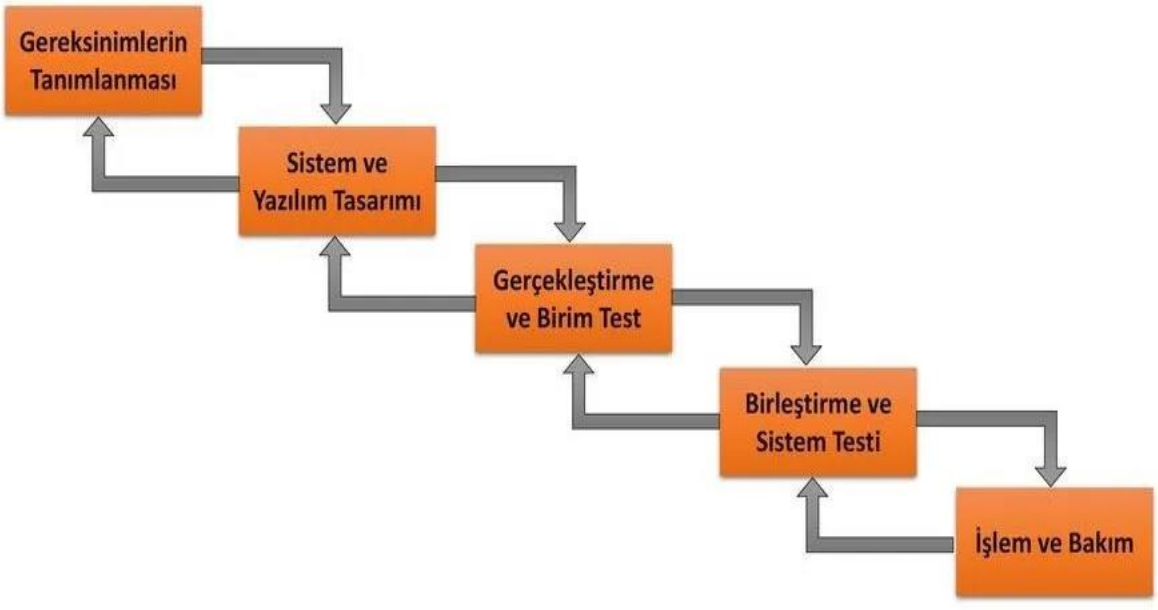
Yazılım geliştirmenin zorluklarıyla baş edebilmek için geliştirmeyi sistematik hale getirmeyi hedefleyen çeşitli süreç modelleri ortaya çıkmıştır. Bu modellerin temel amacı; proje başarısı için yazılım geliştirme yaşam döngüsü boyunca izlenmesi önerilen mühendislik süreçleri tanımlamaktır.

ÇAĞLAYAN MODELİ

Yaşam döngüsü temel adımları baştan sona en az bir kere izlenilerek gerçekleştirilen modeldir. Geleneksel model olarak da bilinmektedir ve günümüzde kullanımı giderek azalmaktadır. İyi tanımlanmış olan ve üretimi az zaman gerektiren yazılım projeleri için uygun bir modeldir. Bir sonraki aşama, önceki aşama tamamlanmadan başlayamaz. Her aşamanın sonucu en az bir defa onaylı olan belgedir. Gerek duyulduğu takdirde geliştirme aktivitelerinde tekrarlamalar olabilir. Bir yazılım yaşam döngüsü olan Barok modelinin aksine belgeleme işlevini ayrı bir aşama olarak ele almaz, üretimin doğal bir parçası olarak görür.

ÇAĞLAYAN MODEL AŞAMALARI

1. Gereksinim tanımlama: Gerçekleştirilecek projenin gereksinimlerinin belirlendiği aşamadır.
2. Sistem ve yazılım tasarımı: Gereksinimleri tanımlanmış bir sistemin yapısal ve detay tasarımının oluştuğu aşamadır.
3. Gerçekleştirme ve birim test: Tasarımı yapılmış olan sistemin kodlanma aşamasıdır.
4. Birleştirme ve sistem testi: Gerçekleştirilmiş olan sistemin istenen işlevselliği gösterip göstermediğinin sınandığı aşamadır.
5. İşlem ve bakım: Teslim edilmiş ürünü değişen ihtiyaç ve isteklere göre yeniden düzenleme aşamasıdır.



ÇAĞLAYAN MODEL AVANTAJLARI

Değişiklik süreci yönetilebilir süreçlere bölündüğü için iş dağılımı yapmak kolaydır. Müşteriler ve kullanıcılar tarafından açık olarak anlaşılabilen adımlardan oluşur. İterasyonlar bir önceki ve bir sonraki adımlarla gerçekleşir. Gereksinim adımından sonraki temel sağlamdır. Erken işin miktarını artırır. Gereksinimleri iyi anlaşılabilen ve kalite gereksinimlerinin, bütçe ve zaman kısıtlamasından daha önemli olduğu projelerde iyi çalışır.

ÇAĞLAYAN MODEL DEZAVANTAJLARI

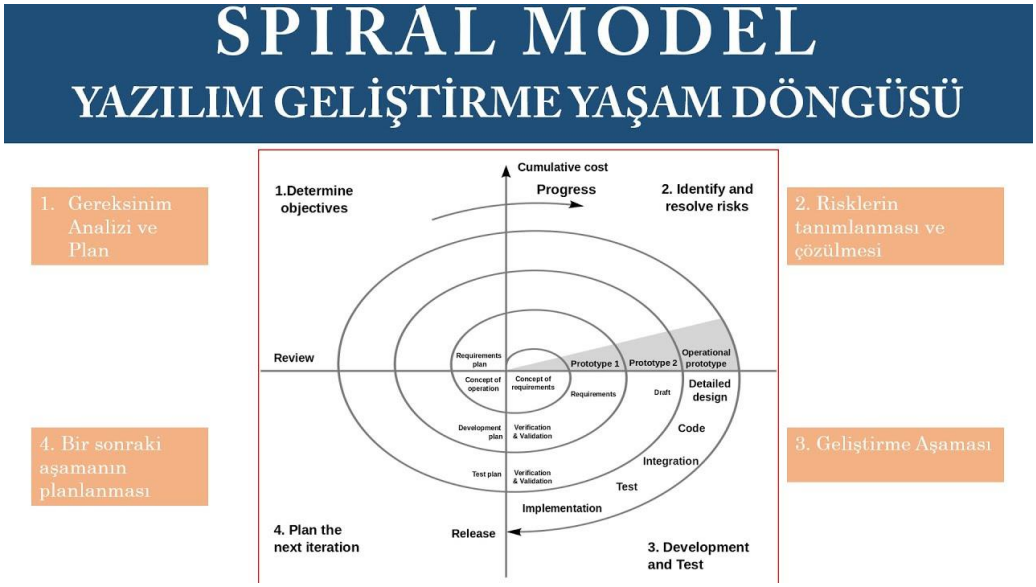
Kullanıcı sürecin içinde olmadığından dolayı teslimden sonra geri dönüşler artabilir ve bu maliyet açısından üreticiyi zarara sokar. Önceki fazlara gitmek oldukça maliyetlidir. Sistem geliştirmesi uzun sürede gerçekleştiği için gereksinimler sürekli değişebilir. Bitirme kriteri olarak belgelendirmeye önem verilmektedir. Yazılımcılar daha çok sonuca odaklı olduklarından dolayı yazılım dışında kalan kriterlere pek fazla önem verilmez.

HELEZONİK MODEL (SPİRAL MODEL)

Bu model, risk analizi ve prototip üretme üzerine kurulmuştur. Her döngü öncesi içinde bulunan fazın risk analizi yapılır ve o faz için uygun olan prototip geliştirilir. Her döngü bir aşamayı ifade eder. Helezonik model, önceden geliştirilmiş yazılım bileşenlerinin yeniden kullanıldığı projeler için çok uygundur. Yinelemeli artımsal bir yaklaşım vardır.

HELEZONİK MODEL AŞAMALARI

1. Planlama: Üretilecek ara ürün için işin planlanması, amaç ve kısıtların belirlenmesi, bir önceki adımda üretilmiş olan ürün ile tümleştirme yapılması faaliyetlerini içerir.
2. Risk yönetimi: Alternatifler değerlendirilir ve risk analizi yapılır.
3. Üretim: Planlanmış ara ürünün geliştirildiği aşamadır.
4. Kullanıcı değerlendirilmesi: Ara ürün hakkında kullanıcıların test ve değerlendirmeleri yapılır.



Spiral üzerinde her bir halka bir fazı gösterir.

Belirtim, tasarım gibi kesin fazları yoktur.

Spiraldeki halkalar neye ihtiyaç varsa onu gerçekleştirmek için seçilir.

Süreç boyunca risklerin değerlendirilmesi ve çözümü açık olarak yapılır.

HELEZONİK MODEL AVANTAJLARI

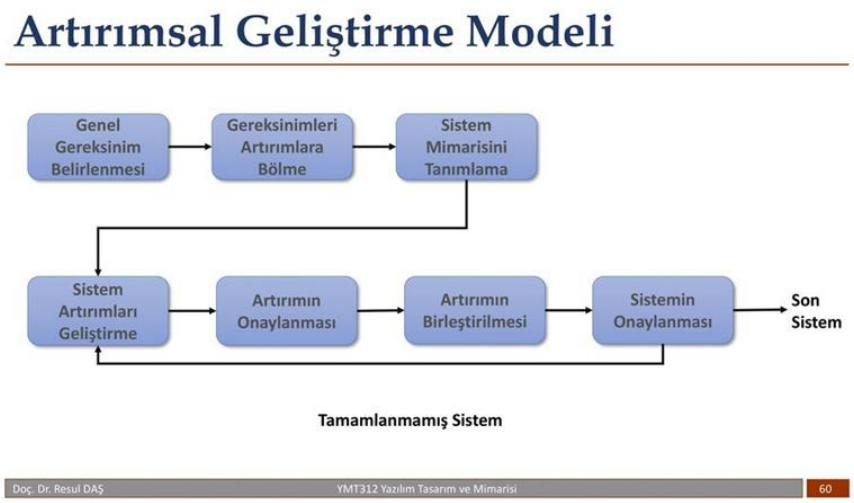
Her döngü başında risk analizi yapıldığı için zaman ve maliyet bileşenleri kolay tahmin edilebilir. Projelerin kaliteye yönelik hedeflerinin önceden belirlenmesi sayesinde bu kalite hedefleri her döngüde alternatif ve kısıtlar belirlendiği için diğer modellere göre daha kolaydır. Kullanıcılar sistemi erken görebilirler. Yazılım-donanım sistemi geliştirme için bir çerçeve sağlar. Birçok yazılım modelini bünyesinde barındırır.

HELEZONİK MODEL DEZAVANTAJLARI

Bu model küçük projeler için uygun bir model değildir. Çünkü küçük ve düşük riskli projeler için maliyetli bir yöntemdir. Spiral sonsuza gidebilir, kompleks bir modeldir. Modeli kullananların tecrübeli olması gerekmektedir. Risk analizi olgusuna dayandığından alt yüklenici kullanımında zorluklar taşır. Kontrat tabanlı yazılıma uymaz. Öznel risk değerlendirme deneyimine dayanır. Ara adımlar fazla olduğu için çok fazla dokümantasyon gerektirir.

ARTIRIMSAL GELİŞTİRME MODELİ

Üretilen her yazılım sürümü birbirini kapsayacak ve giderek artan sayıda işlev içerecek şekilde geliştirilir. Uzun zaman alabilecek ve sistemin eksik işlevlikle çalışabileceği türdeki projeler bu modele uygun olabilir. Bir taraftan kullanım yapılırken diğer taraftan üretim yapılır. Kullanıcı gereksinimlerine öncelik verilir. Öncelikle ürüne ilişkin çekirdek bir kısım geliştirilerek uygulamaya alınmakta ve ardından yeni işlevsellikler eklenerek yeni sürümler elde edilmektedir.



ARTIRIMSAL GELİŞTİRME MODELİ AVANTAJLARI

Sistem için gerekli gereksinimler müşterilerle belirlenir. Teslim edilecek artımlar belirlenir ve bu artımlar gereksinimlerin daha iyi anlaşılmasını sağlar. Tüm projenin başarısız olma riskini azaltır. Böl ve Yönet yaklaşımıdır.

ARTIRIMSAL GELİŞTİRME DEZAVANTAJLARI

Her bir parçanın kendi içinde tekrar etmesine izin verilmez. Bir ara ürün bitip diğeri başlayana kadar herhangi bir değişiklik yapılamaz. Kullanım için tecrübeli personel gereklidir. Gereksinimleri doğru boyuttaki artırımlara atamak zor olabilir.

Günümüze kadar birçok geleneksel yazılım yaşam döngü modelleri denenmiştir. Ancak rekabetin artmasının doğal sonucu olan hızlı istek değişikleri, eski yöntemlerde meydana gelen gecikmeler, bütçe aşımları ve başarısız sonuçlar firmaları "Çevik Proje Yönetimi" ne yönlendirmiştir. Çevik proje yöntemi çok kapsamlı olmayan ve belirsizliğin çok fazla olduğu, ihtiyaçların tam anlamıyla belirlenemediği projeler için kullanışlı bir modeldir. Çevik proje yöntemi uygulamalarından biri de SCRUM tekniğidir.

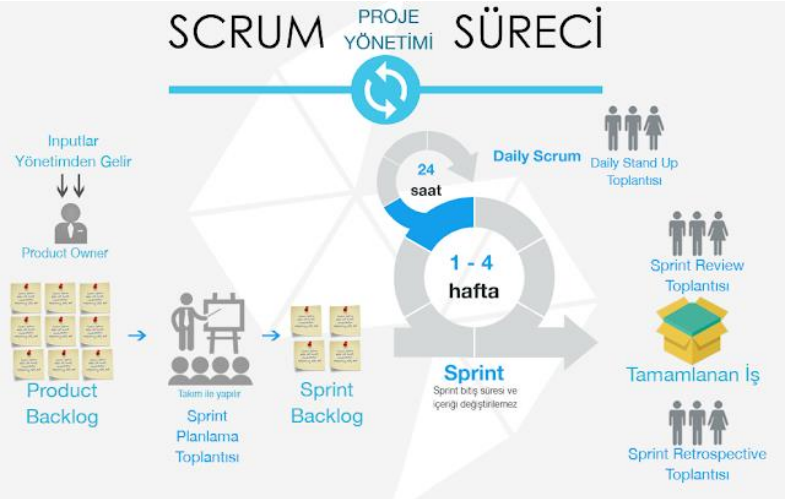
SCRUM TEKNİĞİ

Scrum, basit kuralları olan bir yönetimsel modeldir. Gereksinimleri açıkça belli olmayan, değişime açık, karmaşık yazılım projelerinin yönetimi için uygulanmaktadır. Scrum 1990'ların başında yazılım endüstrisinde daha hızlı ve etkili projeler geliştirebilmek amacıyla Ken Schwaber ile Jeff Sutherland tarafından geliştirilen bir yöntemdir.

SCRUM SÜRECİ

1. Proje sorumlusu geliştirilmesi gereken tüm özellik ve fonksiyonları öncelik sıralarına göre toplayarak Product Backlog (ürün gerikaydı) hazırlar.
2. Takım belirlenerek "Sprint Planning" denilen en fazla 4 hafta sürecek olan küçük döngülerle (sprint) işe başlanır.
3. Her döngü için ürün gerikaydından önemli gereksimler seçilerek sprint gerikaydı oluşturulur ve sprint boyunca geliştirilir.
4. Takım "sprint" boyunca her gün Scrum Master liderliğinde bir araya gelir ve en fazla 15 dakika içerisinde her takım üyesi kendi ilerlemesini kısaca belirtir. Her gün yapılan bu kısa görüşmelere Daily Scrum Meeting denir. Sprint boyunca Burndown Chart denilen kalan gereksinimler/geçen zaman grafiği güncellenir.
5. Sprint bittiği zaman bir "Sprint Review" raporu çıkarılır ve sprint süresince ortaya çıkan sonuçlar ortadan kaldırılarak bir sonraki sprintteki iş yükü azaltılmış olur.
6. Yeni bir sprint için tekrar gereksinimler seçilir ve tekrar sprint döngüsü başlar.

Scrum'da müşterinin üründen beklentileri karşılaması ilk önceliktir. Müşteri de sürecin içinde aktif olarak rol alarak proje gelişimini takip eder. Düzenli aralıklarla ekipler kendi yöntemlerini gözden geçirerek verimliliği artırmak için gerekli iyileştirmeleri yapar.



KAVRAMLAR

1. Product Backlog (Ürün Gereksinim Dökümanı): Proje için gerekli gereksinimler listesidir. “Ne üretilmek isteniyor?” sorusuna cevap aranır. Ürün sahibi tarafından müşteriden gereksinimler alınır, öncelik sırasına göre sıralanır. Değişen ihtiyaçlara göre product backlog’a ekleme-çıkarma yapılır ve bu sayede projenin her aşamasında projeye kolayca entegre edilebilir olur.
2. Product Backlog Item: Product Backlog içindeki her bir gereksinime verilen isimdir.
3. Sprint: Proje sprint denilen küçük kısımlara ayrılır. Scrum içerisindeki tüm aktiviteler sprint içinde gerçekleşir. 1-2 haftalık süreçlerdir.
4. Sprint Backlog: Bir sprint boyunca yapılacak itemlerin listesini oluşturur.
5. Scrum Board: Bir sprint içinde yapılacaklar burada yönetilir.
6. Burndown Chart: Yatay ekseninde sprint günlerini, dikey eksende sprintte kalan işleri gösteren grafiktir.

ROLLER

1. Ürün Sahibi: Geliştirme takımı ve müşteri arasındaki iletişimi sağlar. Projenin özelliklerini tanımlar. Sprint’i iptal yetkisine sahiptir.
2. Scrum Yöneticisi: Scrum kurallarını, teorilerini ve pratiklerini iyi bilir ve takımın bu kurallarını uygulamasından sorumlu kişidir. Verimli çalışmaları engelleyen durumları ortadan kaldırır.
3. Scrum Takımı: Bir Sprint’e alınan bütün işleri tamamlayacak özelliklere sahip kişilerdir. Kendi kendini yönetir. İşin verilmesini beklemezler, işi kendileri alır ve geliştirirler.

Müşteriler ve satıcılar gibi Scrum’ın işleyişinde aktif olarak yer almayan yan roller de vardır.

TOPLANTILAR

1. Sprint Planlama: Product backlog ile belirlenmiş gereksinimler bu toplantıda küçük görevlere ayrılır. Bu toplantıya ürün sahibi, Scrum yöneticisi ve geliştirme takımı katılır.
2. Günlük Scrum Toplantısı: Her gün aynı yer ve aynı saatte ayaküstü yapılan 10-15 dakikalık toplantılardır. Herhangi bir sorun varsa bu sorun scrum yöneticisi tarafından halledilir.
3. Sprint Gözden Geçirme: Her sprint sonunda yapılır. Yapılan sprint gözden geçirilir ve ürün değerlendirilir. Amaç ürünün müşteri gereksinimlerine uygun olarak oluşturulup oluşturulmadığıdır.

SCRUM TEKNİĞİ AVANTAJLARI

Geleneksel yöntemlerde müşteri sadece gereksinim belirleme ve son ürün aşamasında sisteme dahilken Scrum tekniğinde müşteri-kurum ilişkisi söz konusudur. Tüm yazılım geliştirme süreci sprintlere bölünür ve her arada oluşan ara ürün müşteriye sunulur. Böylece geri dönüşler daha hızlı ve etkili hale gelir. Öncelik sırasına konulmuş gereksinimlere odaklanıldığı için fazla işi ve iş yükünü engeller. Müşteri memnuniyeti gibi faktörler organizasyonun pazar potansiyelini artırır.

SCRUM TEKNİĞİ DEZAVANTAJLARI

Belirli bir bitiş tarihi belirlenmediği sürece proje teslim edilene kadar yeni işlevsellikler talep edilmeye devam edilecektir. Ekip yeni başlayanlardan oluşuyorsa aksamalar ve gecikmelerin olması kaçınılmazdır. Ekip üyeleri taahütte bulunmadığı sürece proje tamamlanmaz ya da başarısız olur.

KAYNAKLAR

https://sibelkaldan.medium.com/yazilim-ya%C5%9Fam-d%C3%B6ng%C3%BCs%C3%BC-modelleri%CC%87-ve-scrum-778020af15e1

https://www.ceyrekmuhendis.com/scrum/

https://acikders.ankara.edu.tr/pluginfile.php/89216/mod_resource/content/1/Yaz%C4%B1l%C4%B1mMuh2.pptx