**Yasin Bera Şahin**

**220601069**

**YAZILIM YAŞAM DÖNGÜ MODELLERİ**

**YAZILIM YAŞAM DÖNGÜSÜ NEDİR?**

Geliştirdiğimiz yazılım projesinin planından teslimine kadar geçirdiğimiz sürece ve sırayla; planlama, analiz, tasarım, gerçekleştirme, bakım aşamalarından oluşan döngüye, yazılım geliştirme yaşam döngüsü denir. Bu döngü, yazılım geliştirmenin yanı sıra risk düşürücü çözümler içererek kaliteyi arttırmayı amaçlar.

**YAZILIM YAŞAM DÖNGÜSÜ AŞAMALARI**

**1-Planlama:** Yaşam döngüsünün ilk basamağı olan ve bu proje ile ne yapmayı hedefliyoruz sorusunun cevabıdır. Bu aşamada geliştirilecek sistemin ihtiyaçları belirlenir. Projenin bir taslağı niteliğindedir.

**2-Analiz:** Proje detaylı bir şekilde incelenir ve sorun olan kısımlar tespit edilir. Belirlenen gereksinimler dokümante edilerek “Bu proje ile hedeflediğimiz noktaya nasıl gidebiliriz?” sorusunun cevabını ararız.

**3-Tasarım:** Yazılım projesinin altyapısının oluşturulduğu kısımdır. Analiz aşamasında belirlenen gereksinimleri karşılamak üzere planlama ve analiz basamaklarından da yararlanılarak bir proje çizilir. Tasarım aşaması iki kısımdan oluşur: Mimari tasarım ve detaylı tasarım. Mimari tasarımda belirlenen plan üzerinde modüller, akış şemaları yapılır. Detaylı tasarımda ise yazılım bileşenlerinin ayrıntıları yer alır; veri yapıları, kullanılan dil, ekran tasarımı gibi.

**4-Gerçekleştirme:** Bu aşamada artık modüller, kodlanır, birleştirilir, ve test edilir. Gerçekleştirme aşamasının en önemli kısımlarından biri de testtir. Çünkü bu kısımda “istediğimiz sonuca ulaşabildik mi?” sorusunun yanıtını veririz. Burada mühendis olarak beklenen sonuca ulaşabildiğimizi kontrol ederiz ve olağan bir sorunda erken müdahale yapabiliriz. Bu şekilde hem vakitten hem de para ve prestij kayıplarının önüne geçmiş oluruz.

**5-Bakım ve Destek:** Artık uygulamamız son bulmuş ve müşteriye teslim edilmiştir. Teslimden sonra olan hataları giderme ve geri bildirimleri toplayıp çıkan hataların ve gereksinimlerin giderildiği son aşamadır.

**YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ**

Yazılım geliştirme süreci boyunca birçok farklı türde yazılım geliştirme yaşam döngüsü tasarlanmıştır. Modellerin ortaya çıkmasındaki başlıca nedenler; kalite, dinamik ortam, risk düşürücü çözümler, modüler yaklaşım ve yazılım geliştirmenin zorluklarıyla başa çıkabilmektir. Bu modeller yazılım sektörünün gelişiminde ve ihtiyaçlar açısından önemli rol oynamaktadır. Yazılım sektöründe sıklıkla kullanılan bazı yazılım yaşam döngüleri aşağıdaki gibi açıklanmıştır.

**Çağlayan/Şelale(Waterfall) Modeli**

Çağlayan modeli projenin başarısını sağlamak için yazılım mühendislerinin yaygın olarak kullandığı ilk SDLC modelidir. Bu modelde yazılım geliştirilirken bir aşama tamamlanmadan diğer aşamaya geçilmez ve süreç doğrusaldır. Bu aşamalar analiz, tasarım, kodlama, test, sürüm ve bakımdır. Her aşamanın sonunda dökümantasyon yapılır. Bu sayede kolay anlaşılabilir olmasını sağlar. Projenin aşamaları ayrı olduğundan iş planları projenin en başından bellidir ve bu durum projenin yönetimini kolaylaştırır. Bu projede aşamalar doğrusal ilerlediğinden dolayı projede oluşabilecek bir değişim büyük maliyetlere mal olabilir. Bu yüzden değişime açık bir model değildir. İlk başta müşterinin ne istediğini anlamak ve anlarken de aklımıza takılan en küçük soruları dahil sormamız lazımdır. Aksi takdirde ileriki aşamalarda müşteri istediği projenin tam olarak yapılmadığını söyleyebilir ve gerginliklere yol açabilir.

**V Modeli**

Geliştirme sürecinin “V” şeklinde yürütüldüğü bir yazılım geliştirme modelidir. Adımlar “V” şeklini oluşturduğu için V şeklinde model denilmektedir. Şelale modelinin gelişmiş halidir. Şelale modelinden farklı olarak yazılım geliştirmeye başlamadan test planı oluşturulur. Her bir analiz paralel karşılığı olan testlerden geçerek bir sonraki safhaya ilerler. Bu model doğrusal bir yolla ilerlemek yerine süreç adımlarının kodlama aşamasından sonra yukarı doğru eğim aldığı bir yol izler. Bu yüzden “V” şeklini oluşturur ve adını da buradan alır. V modelinin sol tarafında üretim, doğrulama adımları ve sürecin şartlarının uygunluğuna bakılır. Sağ tarafı ise test işlemleri yani “sistem uygun çalışıyor mu?” daha çok nitel sorulara cevap aldığımız kısımdır. V modeli disiplinli bir modeldir. Her bir aşama paralelindeki test evresinde olumlu ya da olumsuz bir cevap alacağı için kontrollü bir şekilde ilerler. Bu da bize disiplinli bir proje geliştirmeyi sağlar. Belirsizliklerin az olduğu, iş tanımlarının belirgin olduğu projeler için uygundur. Müşterinin gereksinimlerine açık bir modeldir. Müşterinin ne istediğini net bildiği ve projeyi geliştiren mühendisin de sorunun ne olduğunu tam anladığı projelerde kullanılması uygundur. Basit ve anlaşılması kolay bir modeldir. Çünkü test aşamaları bize olumlu veya olumsuz cevap verdiği için bizim nasıl ilerleyeceğimize yol gösterir. V modeli genellikle finansal yazılım alanlarının tercih edildiği bir modeldir.

**Spiral Modeli**

Şeklinden de anlaşılacağı gibi projenin x ve y ekseninden kesilerek dört çeyreğe bölünmesinden oluşan helezon bir proje geliştirme modelidir. Her bir tam tur bir döngüdür. Her döngü öncesinde içinde bulunan fazın analizi yapılır ve o faz için planlanan proje geliştirilir. Her sonunda da yeni planlar yapılarak, hedefler belirlenir. Spiral modelin kesin döngü sayısı olmayıp projenin gidişatına göre değişiklik gösterir. Spiralin yarıçapı o ânâ kadarki projenin maliyetini gösterir. Spiralin her bir açısal boyutu ise o modelin şu ânâ kadar yaptığı ilerlemeyi gösterir. Spiral model dört çeyrekten oluşur. İlk çeyreğinde gereksinim, analiz ve plan aşaması vardır. İkinci çeyrekte risklerin tanımlaması ve çözümlemesi vardır. Üçüncü çeyrek ise geliştirme aşamasıdır. Bu kısımda proje tasarlanır, kodlanır, test edilir ve tamamen uygulanır. Döngünün son aşaması olan son çeyreğinde ise müşteri yazılımın değerlendirmesini yapar. Eğer müşterinin rahatsız olduğu yer olursa mühendis ona göre döngüye devam eder ya da sonlandırır. Bu modelin avantajları büyük ve karmaşık projelerde rol oynar, hataları erken gidermeye odaklıdır ve en riskli kısımlar önce gerçekleştirilir. Dezavantajları ise karmaşıktır, pahalıdır ve kontrat tabanlı yazılıma uymaz.

**Artımsal Geliştirme Modeli**

Yazılımın küçük parçalara ayrılarak döngüsel olarak geliştirilmesidir. Gerekli olan gereksinimler müşteriler tarafından belirlenir. Daha fazla test edilme imkanından dolayı başarısız olma riski azdır. Erken artımlar prototip gibi davranarak, gereksinim daha iyi algılanmasına olanak sağlar. İyi bir planlama ve tasarım gereklidir. Bir maddenin bir tarafında üretim bir tarafında tüketim yapılır. Deneyimli personellerce yapılır.

**Scrum**

Maliyeti yüksek ürünleri üretirken ve yaratıcı bir şekilde geliştirirken karmaşık problemlerin olduğu bir çerçevedir. Scrum, anlaşılması basittir ama profesyonel bir biçimde yönetilmesi oldukça zordur. Scrum, karmaşık ürün ve geliştirilme süreçlerini şeffaflaştırarak denetimi kolaylaştırır. Scrum’da nelerin yapılacağı ile ilgili kısıma “sprint” denir. Sprint’te yapılacak şeyler planlanır. Ortaya hızlı bir şekilde potansiyel bir ürün çıkarılır ve bu ürün gözden geçirilir. Sonunda da sprint’in daha iyi yönetilebilmesi için bir sprint retro toplantısı yapılır. Özetle sprint; planlanır, koşullanır, değerlendirilir ve son olarak süreç iyileştirilmesi yapar. Günümüzde sıklıkla kullanılan scrum’ın popüler olmasının sebepleri ise; görevleri organize bir şekilde yönetip ve bunlara ulaşma noktasında gerekli planlamaya, yüksek donanıma sahip olması, zamandan ve maliyetten tasarruf etmemize olanak sağlar ve kısa sürede yüksek başarı sağlar. Bu yüzden daha fazla tercih edilmektedir.

Yazılım süreç modelleri birbirleriyle ilişkilidir. Birçok model olduğu için faydalı olacağını öngörebiliriz ve buna uygun seçim yapabiliriz. Bu seçimleri yaparken; projenin karmaşıklığı, şirket yapısı, maliyet ve zaman gibi unsurlar da göze alınır. Bu modeller günümüzde planlı, programlı, denetleyici, zamandan ve maliyetten tasarruf sağladığı için sıklıkla kullanılmaktadır. Her açıdan müşteri konforu ve memnuniyetinin ön planda olması, daha etik bir ortam oluşturmaktadır.

**GitHub=> YasinBeraSahin**

**Medium=> Yasin\_Bera\_Şahin**

**Linkedin=> Yasin Bera Şahin**

**KAYNAKÇA**

**[1] https://fikirjeneratoru.com/yazilim-proje-yonetimi-yontemleri/**

**[2] https://furkanalniak.com/yazilim-muhendisligi-yazilim-surec-modelleri/**

**[3] https://tr.wikipedia.org/wiki/Waterfall\_model**

**[4] https://tr.wikipedia.org/wiki/Scrum**