YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ VE SCRUM NEDEN POPÜLER

**Yazılım Yaşam Döngüsü Nedir?**

Kısa ve net hali ile yazılımın ilk düşünülmeye başlamasından emekliye ayrılmasına kadar olan yazılımın izlediği yoldur. Beş temel adımda incelenir.

1.Gereksinimlerin belirlenmesi: Müşteri gereksinimleri belirlenir.

2.Çözümleme: Sistemin işlev ve gereksinimleri ayrıntılı olarak incelenir. Var olan işler ve sorunlar belirlenir.

3. Tasarım: İncelenen gereksinimlere uygun yazılım sistemi iskeleti belirlenir.

4.Gerçekleştirme: Proje kodlanır ve test edilir.

5.Bakım: proje teslim edildikten sonra alınan geri dönüşlere göre değişiklikler yapılır. (Hata giderme ve yeni özellik ekleme)

**Belli başlı Yazılım yaşam döngüsü modelleri**

1.Gelişigüzel: Bir model olarak sayılmaz çünkü bir yöntemi yoktur. Basit tek kişilik projeler harici önerilmez. (Günümüzde kullanım dışı)

2.Barok Modeli: Yaşam döngüsü adımlarını döngüsel değil doğrusal ele alır. Diğer modellerden farklı olarak belgeleme ayrı bir adım olarak ele alınır. (Günümüzde kullanım dışı)

3.Şelale Modeli: Modern yazılım yaşam döngüsü modellerinin atası konumundadır. Bu ve bundan sonraki modellerde belgeleme işin doğal bir parçası olarak sayılmaya başlanır ve ayrı bir adım olmaktan çıkar. Üretim sürecindeki adımlar baştan sona en az bir kere tekrarlanır. Her aşamaya başlamak için bir önceki aşamanın bitmesi gereklidir. Gereksinimleri iyi tanımlanış ve uzun olmayan projeler için uygundur. Gereksinimlerin çok net bir şekilde tanımlanması genelde olmadığından dolayı proje teslim edildiğinde sıklıkla revize gerektirir bu da maliyeti yükseltir. Ayrıca yazılımın son kullanıcıya ulaşması bu döngüde daha uzun sürdüğünden müşteride memnuniyetsizlik oluşturabilir. Bu ve buna benzer sebeplerden günümüzde kullanımı giderek azalmaktadır.

4.V Modeli: Şelale modeline benzer bir modeldir. Şekli itibari ile V şekline benzer. V harfinin sol tarafında yapılan işin çıktısı sağ tarafında sınanır. Bu şekilde hata ayıklama ve programı geliştirme aşamasında kullanıcı etkileşimi artar. V modeli üç alt modelden oluşmaktadır.

-Kullanıcı modeli: Geliştirme sürecinin ile kullanıcı arasındaki ilişkiyi tanımlar ve sınar

-Mimari model: Sistemlerin tasarlar ve sınar.

-Gerçekleştirim modeli: Ürünü kodlar ve sınar.

Belirsizliklerin az ve işin net bir şekilde tanımlı olduğu projeler için uygundur. Modelde kullanıcının projeye katkısı şelale modelinden fazladır

5.Spiral Model: Ara ürünler çıkartarak kullanıcının projeye katkısını arttırmaya yöneliktir. Dört başlığa ayırılır.

-Planlama: Üretilecek ara ürün bir önceki ve bir sonraki ara ürünle uyuşacak şekilde planlanır.

-Risk analizi: Karşımıza çıkacak risklerin belirlenmesi.

-Üretim: Ara ürünün hazırlanması.

-Kullanıcı değerlendirmesi: Ara ürünü kullanıcıya denettirilip geri dönüş alınması.

Risk analizinin ön planda olduğu bu modelde kullanıcı ile etkileşim de fazladır. Ara ürünler çıkartılarak sınandığı için hata miktarı azalır, yöneticiler ve mühendisler elle tutulabilir gelişmeleri sıklıkla gördükleri için motivasyonları yüksek olur. Uzun projeler için uygundur. Ara ürünler sayesinde kullanıcının projeye katkısı diğer modellerden daha fazladır.

6.Artımsal Geliştirme Süreç Modeli: Sistem parçalar halinde teslim edilir. Her teslimde istenen özelliklerin bir parçası teslim edilir. Kullanıcının gerekleri ve gereklerin birbirine bağımlılıkları tespit edilip sıralanır ve teslime en başından başlanır. Bir parça geliştirilmeye başlandığında ek bir gereksinim çıkar ise bir sonraki geliştirme aşamasına kadar bekler. Her sürüm bir önceki sürümü ve daha fazla özelliği içerecek şekilde yapılır ve bir sonraki sürüme eklenecekler için yol gösterici olur. Bir yandan üretim bir yandan kullanım olduğu için uzun vadeli ve sistemin çalışmak için tamamlanmış olması gerekmeyen sistemler için uygundur. Diğer modellerin aksine yazılımın çıkmasından sonra bakım süreci başlamaz. Yazılımın geliştirilmesi devam eder.

7.Kodla ve Düzelt Yaşam-Döngü Modeli: Öncelikle ilk sürüm geliştirilir ve sistem istenen şekle gelinceye kadar devamlı geliştirilir. Yazılım geliştirmenin en kolay ve en pahalı yoludur çünkü sisteme ait belgelendirme olmadığı için bakım ve değişiklik yapmak zordur. Küçük veya kısa ömürlü projeler için uygundur. Yazılımı geliştirmeyi kolaylaştırdığı için küçük veya tecrübesiz firmalar tarafından tercih edilir.

**Scrum Nedir? Yazılım Geliştirirken Nasıl Kullanılır?**

Adını rugby sporundaki tüm takımın birlikte hücum yapması taktiğinden alan scrum yöntemi her alanda uygulanabilir ve uygulanan bir yöntemdir. Büyük işleri küçük parçalara bölüp parça parça bitirmeyi hedefler. Her küçük parçaya koşu denir ve her koşu otuz günü aşamadan bitirilmeye çalışılır ve her gün kısa toplantılarla işin takibi sağlanır. Karmaşık yazılımlar ve istenen ürünün tam belli olmadığı projeler için oldukça uygun bir tekniktir.

**Scrumı daha iyi anlamak için bilinmesi gereken üç temel kavram**

1.Roller

-Ürün sahibi: İstenen ürünü istenilen zamanda ve nitelikte teslim edilmesi ile sorumludur.

-Scrum yöneticisi: Takımı organize etmek ve belirlenen güzergaha sadık kalmaktan sorumludur.

-Scrum takımı: Sürekli iletişimi bulunan ve ortak hedefe ulaşmak için çabalayan beş ila dokuz kişilik takım.

2.Toplantılar

-Koşu planlaması: Gereksinimler tamamı listelenmeye çalışılır. İş küçük parçalara bölünür ve takıma dağıtılır. Riskler değerlendirilir, minimuma indirilmeye çalışılır. Geliştirme araçları ve yöntemi belirlenir. Maliyet hesaplamaları yapılır.

-Koşu gözden geçirilmesi: Her koşu başlarken planlama toplantısı yapılır. Öncelikle gereksinim listesi gözden geçirilir ve listedekilerin içerikleri belirlenir. Sonra scrum takımı gereksinim listesinin başından başlayarak koşunun sonunda bitirilecek olan koşu gereksinim listesini oluştururlar. Bu liste scrum için çok önemlidir

-Günlük scrum toplantısı: Her iş gününde genellikle erken saatte tüm takımın katılımı ile gerçekleştirilen kısa (on beş dakika kadar) toplantıda takımın durumu değerlendirilir. Takımın ilerleyişini ve karşılarındaki sorunları görmek için önemli olan toplantıda tüm takım üyeleri ‘Dün ne yaptım?’, ’Bugün ne yapacağım?’ ve ‘Önümdeki sorunlar neler?’ sorularına cevap arar

3.Bileşenler/Araçlar

-Ürün gereksinimlerinin belgeleri: Proje boyunca yapılmış ve yapılacak iş elemanlarının listesidir. Sıklıkla güncellenen ve bu şekilde kullanışlı kalması sağlanan bu listede her zaman yeni iş eklenebilir, var olan iş silinebilir, var olan işler daha küçük parçalara ayrılabilir ya da farklı işler tek çatı altında toplanabilir.

-Koşu belgeleri: Her bir koşunun koşu belgesi ürün gereksinim belgesindeki işleri kapsar. Koşunun sonucunda istenen ürüne ulaşabilmek için takibi önemlidir.

-Koşu kalan zaman grafiği: yapılması planlanan işlere ne kadar sağdık kalındığını takip etmeye yarar.

**Scrum Neden Popüler?**

Hem elemanlar hem de günler arasında verimli iş dağılımı, günlük iş takibi, karmaşık yapıları basitleştirmesi, takımın iletişimini artırması, verimli bir planlama gibi artıları olmasından dolayı scrum günümüzde yazılım geliştirmenin en popüler yöntemidir.

Kaan Kazguç

İzmir Bakırçay Üniversitesi 210601031

İlgili linkler

<https://medium.com/@kaankzgc/yazilim-ya%C5%9Fam-d%C3%B6ng%C3%BCs%C3%BC-modelleri%CC%87-ve-scrum-neden-pop%C3%BCler-8a866c2a5ed9>

<https://www.linkedin.com/in/kaan-kazgu%C3%A7-804424229/>

<https://github.com/KaanKazguc>

Kaynakça:

<https://iskulubu.com/yazilim/yazilim-gelistirme-yasam-dongusu/>

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

<https://furkanalniak.com/yazilim-muhendisligi-yazilim-surec-modelleri/>

Doç. Dr. Deniz Kılınç Yazılım Mühendisliği Temelleri Ders Notları