Yazılım Yaşam Döngü Modelleri

Azra Nur Akbaba 220601010

Özet

Müşteri ihtiyaçlarındaki değişimlerin hızı artmakla birlikte yazılım projelerinden beklentiler de artıyor. Bu yazının amacı bu beklentilerin karşılanması için oluşturulmuş yaşam döngüsü modelleri kavramını ve bu döngünün içerdiği modelleri incelemektir. Belli başlı yazılım yaşam döngü modelleri hakkında bilgi verilip bazıları detaylı açıklanmıştır. Modellerin açıklanması sırasında temel özelliklerinin yanı sıra bu modellerin birbirlerine göre farkları, avantaj ve dezavantajlarına da yer verilmiş olup çevik yazılım türü olan scrum hakkında bilgiler toplanmış yazıya aktarılmıştır.

Anahtar kelimeler: Yazılım, model, döngü, süreç, gereksinim

Summary

As the pace of changes in customer needs increases, expectations from software projects are also increasing. The purpose of this article is to examine the concept of life cycle models created to meet these expectations and the models included in this cycle. Information about certain software lifecycle models is given and some of them are explained in detail. During the explanation of the models, besides their basic features, the differences, advantages and disadvantages of these models with respect to each other were also included, and information about scrum, which is a type of agile software, was collected and transferred to the article.

Keywords: Software, model, loop, process, requirement

1.GİRİŞ

Yazılım yaşam döngüsü herhangi bir yazılımın üretim ve kullanım aşaması birlikte olmak üzere geçirdiği tüm aşamalar olarak tanımlanır. Yazılım yaşam döngüsü, yazılım işlevleri ve ihtiyaçları sürekli değiştiği ve geliştiği için mutlaka bir döngü

biçiminde düşünülmelidir. Tek yönlü yani doğrusal olarak kesinlikle düşünülmemelidir. [2].

2.YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ

Birçok yazılım yaşam döngüsü modeli olmakla beraber bu başlık altında şunlara yer verilecektir.

- Gelişigüzel model
- Barok modeli
- Çağlayan modeli
- V süreç modeli
- Spiral model
- Kodla ve düzelt modeli
- Artımsal geliştirme modeli
- Evrimsel model

2.1 Gelişigüzel Model

Aslında bir model olarak sayılmayabilir. Gelişigüzel geliştirmede bir model ya da yöntem bulunmaz ve genellikle kişiye bağlı geliştirmede kullanılır. Yazılımın izlenebilirliği ve bakımı zordur.

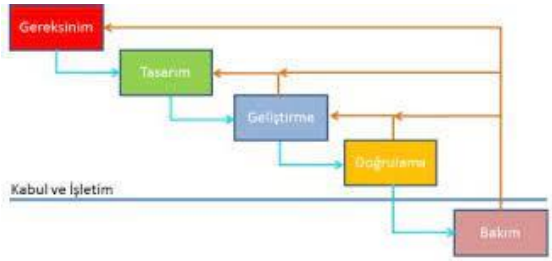
2.2 Barok Modeli

Yazılım yaşam döngüsü temel adımları doğrusal bir şekilde devam eder ancak geri dönüşlerdeki belirsizliklerin nasıl çözüleceği konusunda eksikler vardır. Barok modelinde dokümantasyon basamağı diğer modellerden ayrılarak başka bir basamak şeklinde yer alır.

2.3 Çağlayan Modeli

Diğer bir adı şelale modelidir. Bu modelde çeşitli aşamalar vardır ve her aşama eksiksiz olarak yerine getirilmelidir. Çünkü bir sonraki aşamaya geçebilmek için önceki adımın eksiksiz olarak tamamlanmış olması gerekir. Tamamlanmış kabul edilmesi için de her aşama sonunda dokümantasyon ve test gerçekleştirilmiş olmalıdır. Proje

süresinin uzaması gereksinimlerin değişmesine yol açar. Eskiden kullanımı fazla olsa da günümüzde fazla tercih edilmemektedir.



[1]

Avantajları:

- Anlaşılması kolay şekildedir.
- İş bölümü ve iş planının yapılması gerçekleştirilmesini kolaylaştırır.
- Küçük projelerde kullanımı kolaydır.

Dezavantajları:

- Gereksinim tanımlamaları yanlış yapılırsa projenin ilerlemesi durumunda bu hataların düzeltilmesi zaman alır.
- Kullanıcı sürecin içinde olmazsa proje bittikten sonra geri dönüş olması kaçınılmazdır. Geri dönüşler maliyet artımına ve zaman kaybına neden olur.
- Kodu yazanlar bir an önce bitmesi ve sonuca ulaşılmasına meyilli oldukları için morali düşürmekte geri kalan adımlara gereken önem verilmemektedir.

2.4 V Süreç Modeli

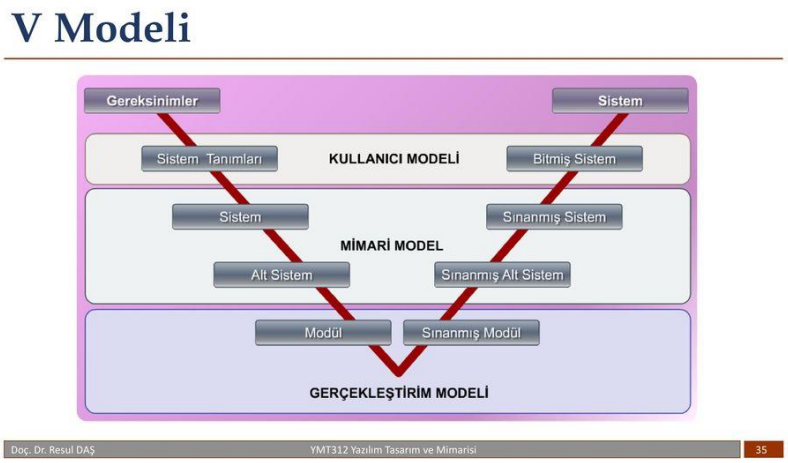
Süreçleri doğrusal bir yol izlemez ve belirli adımlardan sonra tipik olarak V şeklini alarak ilerler [8]. Üç alt modelde inceleyebiliriz. Bunlardan ilki olan kullanıcı modelinde geliştirme aşamasının kullanıcı ile olan ilişkileri tanımlanmakta ve sistemin nasıl kabul edileceğine ilişkin sınama belirtimleri ortaya çıkarmaktadır. Mimari modelde sistem tasarımı ve oluşacak alt modellerin sınama işlemlerine yöneliktir. Gerçekleştirim modeli ise kodlama işlemlerine ilişkin fonksiyonlarda kullanılır. Belirsizliklerin az, iş tanımlarının belirgin olduğu projeler için uygun bir modeldir. [2]

Avantajları:

- V süreç modeli kullanıcının proje içindeki katkısını arttırmaktadır.
- Bir projeyi iki aşamalı olarak uygulamaya koyabilir
- Yinelemelerde ve geri dönüşlerde müşteri ile iç içe olması test sonrası oluşabilecek hataların önüne geçilebilir.

Dezavantajları:

- Fazlar arasında tekrarlama yoktur.
- Risk çözümleme aktiviteleri yoktur.
- İş ve ürün gereksinimleri değişiklik gösterebilir. [2]



[4]

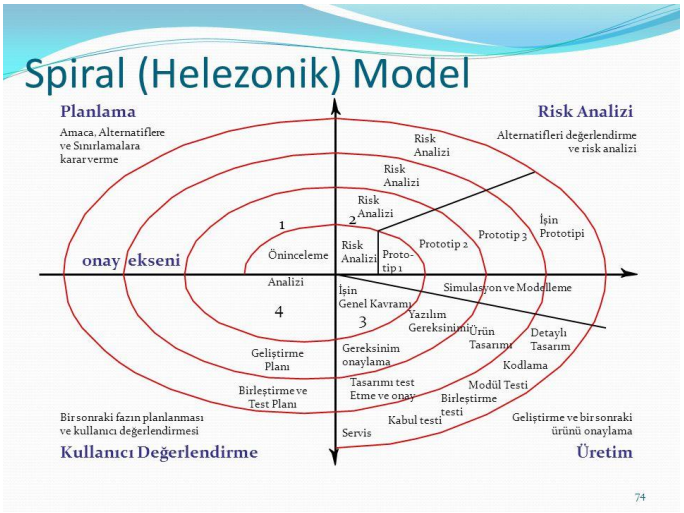
2.5 Spiral Model

Bu modelin en belirgin özelliği risk analizinin ön planda olmasıdır. Prototip ve yinelemeli bir yaklaşıma dayalıdır. Döngülerin her biri bir fazı belirtir. İçinde olunan fazın risk analizi yapıldıktan sonra o faz için planlanmış olan prototip geliştirilir.[5]

Avantajları:

- Her döngü başında risk analizi yapılması sayesinde zaman ve maliyet unsurları erken tahmin edilir.
- Tekrarlanmalar hataların erken fark edilmesini sağlar.
- Diğer bazı yazılım modellerini içinde barındırır.

- Risk analizine önem verilmesi zorlukları engelleyip işi kolaylaştırır.



Dezavantajları:

- Komplekstir. (Karmaşık)
- Küçük projeler için maliyeti arttırabilir.
- Uzun sürer.
- Fazla dokümantasyondan oluşur.
- Spiraller sonsuza gidebilir.

2.6 Kodla ve Düzelt Modeli

Bu model genellikle resmi olmayan bir ürün fikriyle başlar ve program ürün "hazır" olana kadar ya da gerekli zaman bitene kadar kodlama yapılarak devam eder [1]. Teslim sonrası bakım aşaması vardır ve daha sonrasında ise yazılım emekliliğe ayrılır.

Avantajları:

- Planlamaya ihtiyaç duyulmaz.
- Diğer modellere kıyasla en kolay yazılım geliştirme yoludur.
- Büyük tecrübelere gerek yoktur, çoğu şirket ve kişi kullanabilir.

Dezavantajları:

- Bakım zordur.
- Bitiş süresi belli değildir.

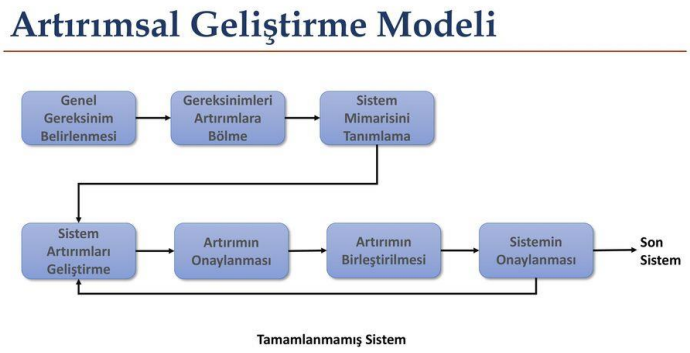
- Maliyeti fazla olabilir
- Kaynak planlaması yapılmaz.
- Disiplinsizdir.
- Doküman yoktur.

2.7 Artımsal Geliştirme Süreç Modeli

Artımsal geliştirme süreç modelinde proje parçalara bölünerek bitirilir ve ürün teslimi bütün şeklinde olmaz. Parçaların sıralaması müşterinin isteklerine bağlıdır yani müşteri gereksinimlerini karşılıyor.

Avantajları:

- Gereksinimler sıraya konulur.
- Öncelikli gereksinimleri müşteri seçer.
- Projenin başarısız olma olasılığını düşürür.
- Divide and Conquer (Böl ve Yönet) yaklaşımıdır.
- Önemli sistem özellikleri test edilme olanağı bulmuş olur.



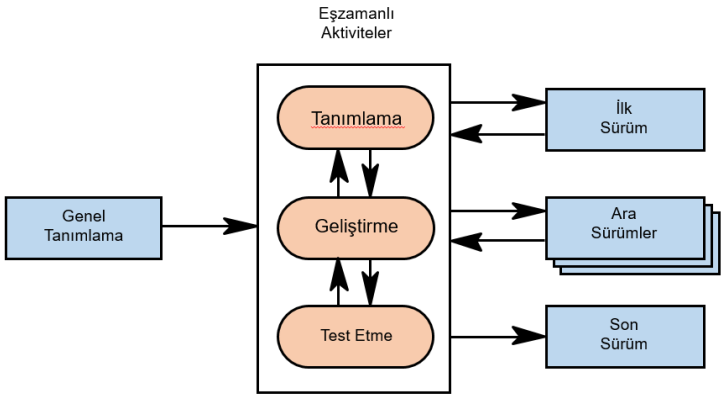
[6]

Dezavantajları:

- Bir ara ürün bitene kadar diğeri başlayana kadar değişiklik yapılmaz.
- Sabit fiyat sözleşmeleri için uygun değildir.
- Deneyimli personel gerektirir.
- Artımların kendi içinde tekrar etme özelliği yoktur.

2.8 Evrimsel Model

İlk tam ölçekli modeldir. Coğrafik olarak geniş alana yayılmış, çok birimli organizasyonlar için önerilebilir buna banka uygulamalarını örnek verebiliriz. Her aşamada üretilen ürünler, üretildikleri alan için tam işlevselliği içermektedirler. Pilot uygulama kullan, test et, güncelle diğer birimlere taşı gibi devam eden bir yapıdır. Modelin başarısı ilk evrimin başarısına bağımlıdır. [4]



[1]

Avantajları:

- Kullanıcıların kendi gereksinimlerini daha iyi anlamalarını sağlar.
- Sürekli değerlendirme erken aşamalardaki geliştirme risklerini azaltır.

Dezavantajları:

- Bakımı zordur.
- Sistemler sıklıkla iyi yapılandırılmaz (sürekli değişiklik yazılımın yapısına zarar verir)

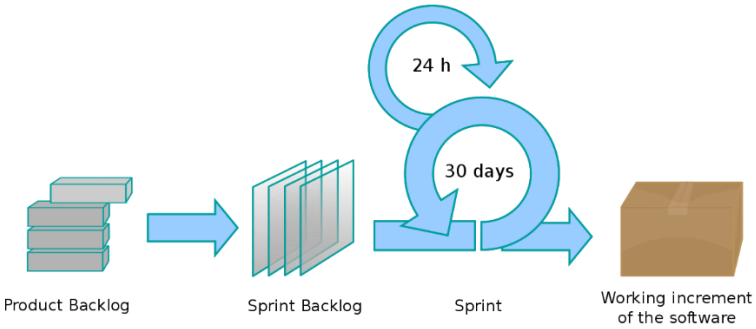
2.ÇEVİK YAZILIM GELİŞTİRME METODOLOJİSİ SCRUM

Yukarıda bahsedildiği üzere bu yaşam döngü modellerine rağmen hala beklentiler karşılanamamış olup bu kötü yanları nedeni ile çevik adıyla yeni metodlar oluşturulmuştur. Çevik yazılım geliştirme yaşam döngü modellerinde karşılaştığımız zaman kaybı, maliyet fazlalığı, yüksek hata oranı gibi problemler için çözüm sağlamıştır. Özelliklerine değinecek olursak çevik yazılım metotları süreç ve ürünlerden çok ekibindeki kişilere önem verir, proje sonuna kadar müşteri ile çalışmayı ister yani ayrılmış olan her iterasyon sonunda müşteriye bilgi verilmelidir, projenin parçalardan

oluşması hataların kısa sürede çözülmesine yardımcı olmaktadır. Scrum kısaca bahsetmiş olduğumuz çevik yazılım metodolojilerden sık kullanılanlardan biridir.

2.1 Scrum nedir?

Scrum aslında rugby'deki bir hücum tekniğinin adıdır. Topu karşı sahaya tüm oyuncuların senkronize desteği ile ulaştırmayı amaçlar. Tıpkı devre aralarındaki taktiksel konuşmalar gibi scrum yönteminin de günlük kısa toplantıları vardır [9]. Her projeye uygulanabilir olan bir proje yönetim yaklaşımıdır. İki ya da dört haftalık sprintler halinde geliştirmeyi merkeze almaktadır. Tercihen büyük ve karmaşık yazılım süreçlerinde kullanılır. Karmaşık yazılımları çözmek için 3 temel prensibi kullanabilir: şeffaflık, denetleme, uyarlama.



[7]

2.2Scrumın temelini oluşturan üç kavram

1.Roller: Ürün sahibi, scrum yöneticisi ve scrum takımları rollere dahildir. Proje yöneticisi rolü bulunmaz ve bulunmaması deneyimsiz ekiplerde soruna yol açabilir bunun çözümü scrum uzmanından yararlanmaktır. Yan rollere ise işletme, müşteri ve kullanıcıyı örnek verebiliriz. Takım kendi içinde 3 yetenek barındırır: yazılımcı, testçi ve tasarımcı.

2.Toplantılar: Sprint planlama ile gereksinimler belirlenir sprint gözden geçirme ile takımlar listeleri oluşturur. Daha sonrasında günlük toplantılar ile takımın ilerleyişi ve karşılaşabilecek engeller belirlenir.

3.Bileşenler/araçlar:

Product backlog: üründe olması gereken tüm işlerin listelendiği, öncelik sırasını gösteren kaynaktır.

Sprint backlog: Sprintler boyunca yapacağı işlerin içeriğini tutan liste diyebiliriz.

Scrum daily meeting: Scrum takımıyla yapılan günlük 30 dakikayı geçmeyen toplantılardır.

4.SONUÇ

Çalışma kapsamında yazılım yaşam döngü modelleri aslında benzer ve farklı yöntemler içermekte ve bir projede birden fazla yöntem kullanılabilmektedir. Gelişigüzel ve barok modelleri artık kullanımını yitirmiştir bunun nedenleri çok dokümantasyonu içermemeleri olabilir. Çağlayan modelini kullanması kolay olmasına rağmen büyük projeler için uygun değildir. Karşılaştırma yapacak olursak Çağlayan yaşam döngü modeli, Barok modelinden farklı olarak dokümantasyonu ek bir süreç olarak ele almaz. Üretimin doğal bir parçasıymış gibi davranır. Barok modelinin aksine bu modelde geri dönüşlerin nasıl yapılacağı da tanımlıdır. Artımsal model gibi davranarak bir projeyi parçalar ve parçaların üzerinde tekrar tekrar çalışılmasını esas alır. Scrum karmaşık yazılım projeleri için kullanılması en uygun yöntemdir.

REFERANSLAR

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

[2] Güven, Z.A., (2023), "Yazılım Mühendisliği Temelleri ders notları", Bakırçay Üniversitesi

[3] https://ybsansiklopedi.com/wp-content/uploads/2015/08/Yaz%C4%B1l%C4%B1m-Geli%C5%9Ftirme-Modelleri-Yaz%C4%B1l%C4%B1m-Ya%C5%9Fam-D%C3%B6ng%C3%BCs%C3%BCSDLCYBS.pdf

[4] http://slideplayer.biz.tr/slide/2806563/

[5]http://www.aspmvcnet.com/tr/m/yazilim-muhendisligi/helezonik-tasarim-spiral-tasarim.html

[6] https://www.google.com/url?sa=i&url=https%3A%2F%2Fslideplayer.biz.tr%2Fslide%2F12386328%2F&psig=AOvVaw0c_z16uC5r8Pee7FVSxaRi&ust=1679741282128000

&source=images&cd=vfe&ved=0CBAQjRxqFwoTCMjg542y9P0CFQAAAAAdAAAAABAE

[7] https://www.google.com/url?sa=i&url=https%3A%2F%2Ftr.wikipedia.org%2Fwiki%2FScrum&psig=AOvVaw0QPvT5SzTiTEJ3dDZBLuCG&ust=1679741550867000&source=images&cd=vfe&ved=0CBEQjhxqFwoTCPiO9I2z9P0CFQAAAAAdAAAAABAE

[8]https://tr.wikipedia.org/wiki/V-Model_(Yaz%C4%B1l%C4%B1m_geli%C5%9Ftirme)

[9]https://medium.com/@fatihazir/yaz%C4%B1l%C4%B1m-ya%C5%9Fam-d%C3%B6ng%C3%BC-modelleri-ve-scrum-a17aef990c4c

Bilgilerim:

https://github.com/Azraakbb

https://medium.com/@azraakbaba

https://www.linkedin.com/in/azra-akbaba-724a8b259/