Merhaba arkadaşlar bugün size yazılım yaşam döngüsündeki modeller hakkında fikirler sahibi olucağınız bilgiler kazandırmaya çalışıcam. İlk olarak yazılım yaşam döngüsü nedir ne işe yarar onu kısaca açıklayım. Bu sayede konumuzu daha rahat bir şekilde anlamanızı sağlamaya çalışacağım.

Nedir bu yazılım yaşam döngüsü denen şey? **Yazılım yaşam döngüsü**, herhangi bir yazılımın, ihtiyaç durumundan gerçekleştirilmesi ve sonrasında bakımına kadar geçen tüm süreçtir. Kısa ve özet kelindeki bilgimizi aldığımıza göre şimdi ise asıl konumuz olan bu bu dögüyü daha efektif kullanabilmek için geliştirilmiş modelleri incelemeye geçebiliriz.

Yazılım yaşam döngüsünde birçok model vardır. Bu modeller projelerin karışıklığını hafifletip karmaşıklıkları önler . Yazılım Şirketleri projenin amaçlarına vara bilmesi için geliştirilmiştir.Bir değil de birden çok modelin olmasının nedenleri olarak yazılım projesinin büyüklüğü,yazılım porjesinin kimler için kullanılıcağıdır. Bunların cevabı verildikten sonra yazılım için uygun model seçilir ve başlanır. O halde şimdi bu modelleri teker teker inceleyelim.

GELİŞİ GÜZEL MODEL

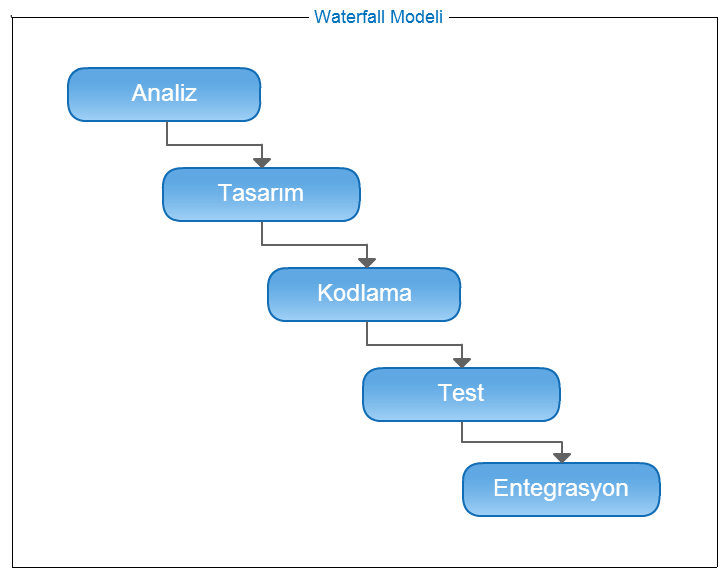
60’lı yıllarda çıkmış bir modeldir.Bir yöntem veya model yoktur.Geliştiren kişiye bağlıdır.Gelişigüzel model genelllikle tek kişilik projelerde ve fazla karmaşık olmayan yazılım sistemlerinde kullanılan bir modeldir.Yani basit programlarda işlevsel olan bir modeldir.Bu arada izlenebirliği ve bakımı oldukça zor bir modeldir.Bu basit model hakkındaki bu kadar bilgi yeterli olduğunu düşünüyorum diğer modelimize geçelim.

BAROK MODELİ

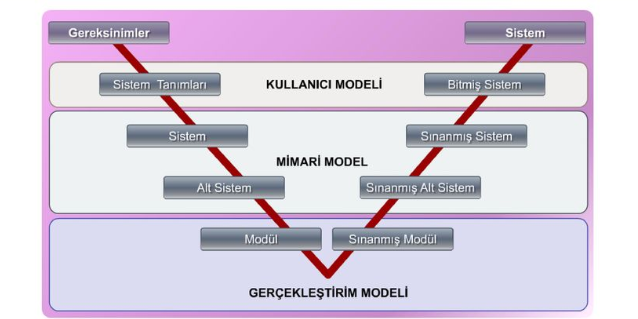
Miadı 70’li yıllara dayanan eski bir yazılım yaşam döngüsü modelidir.Yaşam döngüsünün temel adımları doğrusaldır ve aşamalar arasında nasıl geri dönüş yapılacağı hakkında belirsizlik hakimdir.Bu modelin en önem verdiği aşama ise gerçekleştirim aşamasıdır.Bu modelde dökümantosyon yazılan programn test aşaması bitirilince oluşturulur.Günümüze göre eksisi budur günümüzde her aşama bi dökümantosyon aşaması vardır.Bu gibi çeşitli sebeplerden dolayı günümüz projelerinde tercih edilmeyen bir model haline gelmiştir.

ŞELALE(WATERFALL) MODELİ

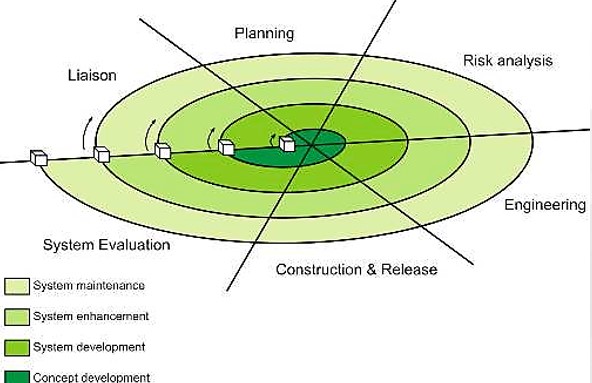
Şelale yönteminde (yaygın kullanılan adı WaterFall Model) yazılım geliştirme süreci analiz, tasarım, kodlama, test, sürüm ve bakım gibi safhalardan oluşur. Geleneksel yazılım metotlarında bu basamaklar şelale modelinde olduğu gibi doğrusal olarak işler. Her basamak, başlangıç noktasında bir önceki basamağın ürettiklerini bulur.  Kendi bünyesindeki değişikler doğrultusunda teslim aldıklarını bir sonraki basamağın kullanabileceği şekilde değiştirir.Barok modeline göre geri dönüşer daha iyi tanımlanmıştır.Barok modelinden farklı olarak dökümantosyon işlemini ayrı bir işlem olarak ele almaz ve her basamağın doğal bir parçası olarak görür.Yazılım tamamlamada belirsizlik yok (ya da az) ise bu modeli kullanmak uygundur. Gerçek yaşamdaki projeler genelde yineleme gerektirir bu yüzden bu modelde genelde yazılımın kullancıya ulaşımı uzundur.Bu modelde yapılan hatalardan birisi ise gereksinimlerin birçok kez net bir şekilde yapılamadığından dolayı,hataların düzeltilme ve eksik noktaların giderilme maliyetleri oldukça yüksektir.Yazılım üretim ekipleri bir an önce programı yazma, progamı çalıştırma ve ürünü görme eğiliminde oldukları için, bu model ile yapılan üretimlerde ekip mutsuzlaşmakta ve kod yazımı dışında kalan zamanda( iş yükğünün %80’ini içeren) kesime önem vermemektedirler.Üst düzey yönetimlerin ürünü görme süresinin uzun oluşu, projenin bitmeyeceği ve devamlı gider merkezi haline geldiği düşüncesini yaygınlaştırmaktadır.Şimdi bu modelin kısaca artı ve eksilerini düzenleyelim.Avantaj olarak şunları söyleyebiliriz birimlere bölünmüştür yani sistemli ve düzenlidir.Bir diğer avantaj olarak gereksinim adımı tamamlandıktan sonra yazılımın temeli atılmış olur.Son avantaj olarak da şunu söyleyebiliriz erken iş yapılır.Gelelim dezavantajlarına öncelikle projede bir aksaklık bulunduğunda fazla geri dönüş bulunduğundan iki ya da daha fazla geri dönüşerlin maliyeti çok fazladır.Diğer bir dezavantaj olarak da bir basamak bitmeden diğer basamağa geçilmesinin riskli olduğu söylenebilir.Bu dezavantajlardan kaynaklı olarak ve yeni güncel modellerin artmasıyla geleneksel model olarak da bilinen bu modelin kullanımı günümüzde giderek azalmaktadır.Bu modelimizi de başarıyla anlattığımıza göre yeni modelimize geçebiliriz.Bu arada Şelala modelinin basamakları aşağıdaki resimdeki gibidir.



V SÜREÇ MODELİ

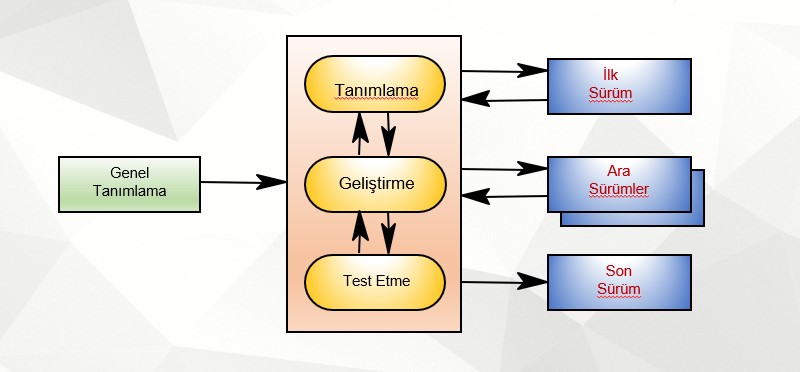
Yazılım yaşam döngü modellerinden olan V modelinde adından da anlaşılacağı gibi “V” şeklinde bir yol izlenir ve bu yüzden de bu modele V süreç modeli denir.V modeli biraz önce anlatığımız Şelale modelinin gelişmişi olarak da düşünebiliriz.Belirsizliklerin az, iş tanımlarının belirgin olduğu BT projeleri için uygun bir modeldir.Model kullanıcının projeye katkısını arttırmaktadır.Aşağıda V modelin şeması verilmiştir. Sol taraf üretim, sağ taraf ise sınamadır.Bu döngü esas olarak 3 modelden oluşmaktadır.Kullanıcı Modeli,Mimari Model,Gerçekleştirim Modeli.Hemen kısaca bu 3 modele de açıklama getirelim.Kullanıcı modelinde kullanıcının istekleri ve proje için tanımları kullanılır ve tamamlanmış hali teslim edilir.Mimari modelde projenin tasarımı ve bunların deneme işlemlerinden oluşur.Gerçekleştirim modeli ise kodlama ve bunların denenmesinden oluşur.Bu modelimizdeki sorunlar ise şunlardır: Aşamalarda tekrar bulunmaz ve risk çözümleme için ayrılan bir bölüm yoktur.

HELEZONİK(SPİRAL MODEL)

Tasarımı doğrusal bir süreç olarak gören modellerin aksine ,bu model spiral bir süreç olarak görür.Bu yineleyici tasarım döngülerini genişleyen bir spiral olarak temsil ederek yapılır.Genellikle iç çevrimler, gereksinim tanımının rafine edilmesi için prototipleme ile birlikte ihtiyaç evresinin erken evresini ve dış spiraller yazılım tasarımını aşamalı olarak temsil eder.Her helezonda, tasarım çabalarını ve bu yineleme için ilgili riski değerlendirmek için bir risk değerlendirme aşaması vardır.Her spiralin sonunda, mevcut spiralin gözden geçirebilmesi ve bir sonraki aşamanın planlanabilmesi için gözden geçirme aşaması vardır.Hemen aşağı sizin için bu modelin gösterimini bırakıyorum.

Her tasarım sarmalının altı ana faaliyeti altı temel görevle temsil edilmektedir bu altı görev sırasıyla:Müşteri iletişimi,planlama,risk analizi,yazılım tasarımı,üretim-dağıtım,müşteri onayı.Şimdi ise bu modelin avantaj ve dezavantajlarını saymaya geldi.Öncelikle avantajlarından başlayalım öncelikle risk analizi yapılmaktadır bu sayede daha kontrollü bir iş ortaya çıkmış olur ve bu yazılım tasarım modeli büyük yazılım projelerini tasarlamak ve yönetmek için diğer modellere göre daha uygundur.Şimdi ise artık dezavantajlarından konuşalım risk analizi yapılıyor demiştik bu risk analiai yüksek uzmanlık gerektiren bir iştir.Bu model kullanması pahalı bir modeldir ve küçük projeler için uygun değildir.Bu model hakkında bu kadar bilginin yeterli olduğunu düşünüp yeni modelimize geçelim.

EVRİMSEL GELİŞTİRME SÜREÇ MODELİ

Bu model ilk tam ölçekli modeldir.Coğrafik olarak geniş alana yayılmış,çok birimli organizasyonlar için önerilmektedir. Modelin her aşamasında üretilen ürünler,üretildiği alan içerisinde tam işlevselliği içermektedirler.Bu modelin basamakları ise pilot uygulama kullan,test et,diğer birimlere taşı.Modelin başarısı ilk evrimin başarısına bağlımlıdır.Bu modelin eksiklerine geçelim.Değişiklik denetimi yoktur.Ve sonrasında konfigürasyon yöntemi de yoktur.Konfigürasyon yönteminin olmamasını açalım.Konfigürasyon yönteminin içinde barındırdığı 3 şey sürüm yönetimi,değişiklik yönetimi ve kalite yönetimidir.Aşağıda bu modelin şeması gösterilmiştir.

ÇEVİK MODELLER

Çevik modeller yazılım projelerindeki başarı oranlarını ve ekip içi iletişimi arttırmak amacı ile ortaya çıkmıştır.Şimdi gelin bazı çevik yazılım geliştirme modellerine bakalım.

EXTREME PROGRAMMİNG(XP)

Kent Back ve arkadaşları tarafından 1996 yılında kurulmuştur.4 temel maddeden oluşmuştur:basitlik cesaret,geri dönüş,iletişimdir.Basitlik yazılan kodun ve yapılan işin sade,anlaşışırve karmaşık omadan yapılmasını gerektirir.Uzun uzun dökümantasyandan uzak durulur.Yapılan işte cesur olunmalıdır,projelerin üstüne korkmadan ilerlenmelidir.Bir kodun gerekirse tamamen silinip yeniden yazılması sağlanmalıdır.En ufak hatadan bile korkulmamlıdır.Geri dönüşler ile hatalar azalır.Müşteri ile yazılım ekibi birbirleriyle sıkı diyolog içindedir.İletişimin projelerde çok büyük bir önemi vardır.XP ekibi içindeki iletişime büyük önem verir ve bu iletişim yükseltmek için çalışır.

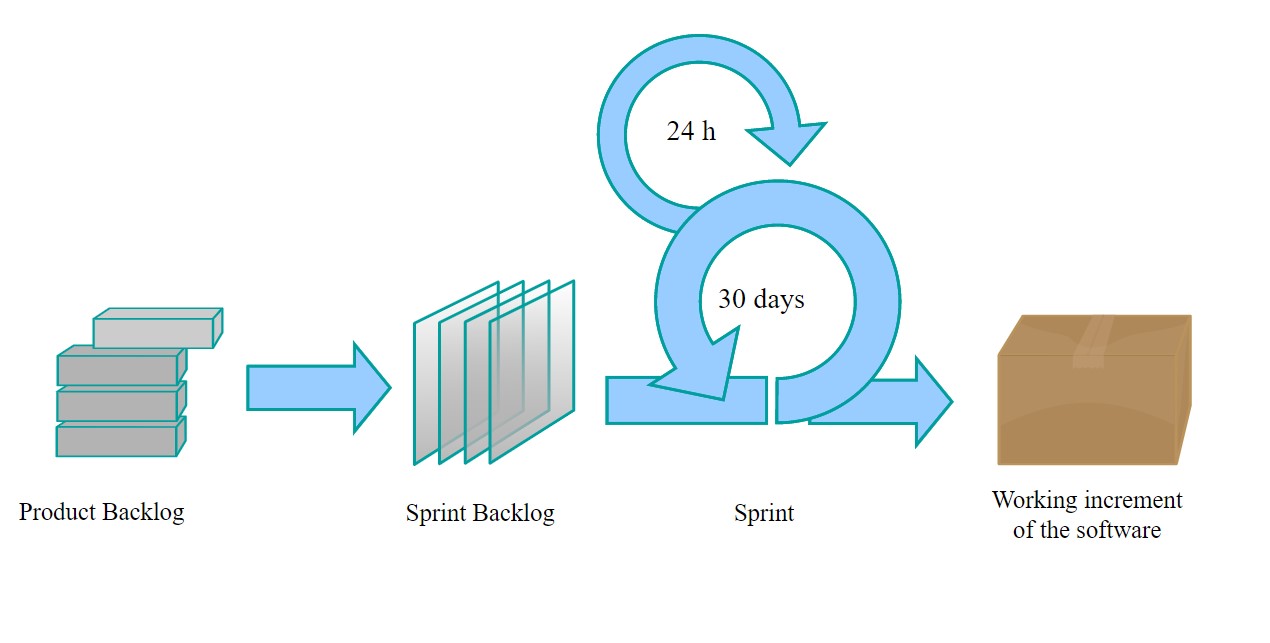
SCRUM

Jeff Sutjerland ve Ken Schawaber tarafından 1990’ların ortalarında geliştirilmiştir.Günümüzde en popüler yazılım geliştirme modellerinden biridir.Temel olarak yazılım ekibinin ve üretilecek yazılımdan faydalanacakların(müşteri) belirli bir süre (genelikle 1 ay) içerisinde maksimum verimlilikte çalışmalarını tüm yazılımın ise bu 1 aylık döngülerden oluşan daha büyük bir proje sonucunda üretilmesini hedefler.Güncelliğini koruyan SCRUM teknolojiyi takip eden birçok şirkette rastgelebileceğiniz bir süreçtir veya modeldir.

Scrum’ın süreçlerini kısaca ifade etmekte fayda var. Bu bağlamda product backlog, sprint backlog, sprint ve scrum toplantılarına kısaca değinelim. Scrum projelerinde ilk yapılan proje sponsorları ile bir masaya oturulup projenin kapsamını ifade eden “product backlog”u çıkartmaktır. “Product backlog” proje sonunda üretilmesi beklenen sistemin özellik ve fonksiyonlarının önceliklerine göre yazıldığı bir liste dokümanıdır. Bu dokümanda, sponsorun ihtiyaçları ve istekleri öncelik sırasına konarak listelenir. Proje süresince bu doküman proje sponsoru tarafından sürekli güncellenir; yeni fonksiyon ve özellikler listeye eklenir, mevcut fonksiyon ve özelliklerin öncelikleri değiştirilebilir.

Scrum yöntemi ile yönetilen projeler “sprint” denilen fazlara ayrılır. Herbir “sprint”in süresi genellikle (en fazla) 30 gündür. İdeal olan proje boyunca boyunca tüm “sprint”ler aynı süreli olur. Proje takımı (scrum team), proje yöneticisi (scrum master) ile her sprint başlangıcında masaya oturarak yeni “sprint” dahilinde geliştirilecek özellik ve fonksiyonları belirler. Bu süreçte seçilen fonksiyon ve özellikler daha önceden oluşturulmuş ve sürekli güncellenen product backlog listesindeki sıra ile en öncelikli olanlardır. Hiçbir zaman önceliği düşük bir özellik veya fonksiyon önceliği yüksek bir özellik veya fonksiyondan önce geliştirilemez. Bu bağlamda bazı “sprint”lerde proje takımı, product backlog’dan 4 eleman, bazı sprint’lerde 25 eleman seçebilir. Seçilen özellik ve fonksiyonlar sprint backlog denilen ikinci bir listeye aktarılır. Proje takımı bir sonraki sprint başlangıcına kadar bir daha product backlog’a bakmaz, o sprint dahilinde sadece ilgili sprint backlog listesine odaklanır. Sprint backlog dahilindeki her özellik veya fonsksiyon için maksimum 3 günlük geliştirme süresi verilir. bu sürede sözkonusu elemanın geliştirmesinin tamamlanmasına çalışılır. Scrum takımı her sabah scrum master eşliğinde “scrum” denilen günlük toplantısını yapar. bu toplantıya tüm takım üyeleri katılmak zorundadır. Scrum toplantısına proje takımı dışındaki diğer proje paydaşları da katılabilir; ancak onların toplantı sırasında kesinlikle konuşma hakları yoktur; sadece dinleyici olabilirler. Konuşan paydaşlar bir sonraki “scrum” toplantısına alınmaz. Scrum toplantılarında kimse oturmaz; herkes ayaktadır. Scrum toplantısında bir önceki gün yapılanlar, o gün yapılacaklar, karşılaşılan ve karşılaşılabilecek riskler tartışılır. Scrum toplantısı en fazla 15 dakika sürer.

Her sprint sonunda sprint raporu çıkarılır. Bu rapor dikey ekseni kalan sprint backlog’da bulunan ve daha geliştirilmemiş özellik ve fonksiyon sayısını, yatay ekseni sprint gün sayısını gösteren iki eksenli basit bir rapordur. Her sprint için bir adet sprint raporu çıkarılır ve günlük olarak scrum master tarafından güncellenir. Sprint’in ilk günü yatay eksendeki değer en yüksektir, sprint sonuna doğru aşağıya doğru eğimli bir çizgi oluşur.



Belirttiğimiz gibi Scrum günümüzde en çok kullanılan yazılım geliştirme yöntemidir.Aslında sadece yazılım geliştirme değil birçok sistemin geliştirilmesinde de kullanılır.Artık Scrum’ın avantajlarına geçebiliriz.Zamandan ve paradan büyük ölçekte tasarruf edilir.Yüksek teknolojiler ve son gelişmelere kolaylıkla uyum sağlar.Karmaşık görülen ve gereksinimleri tam belirlenmemiş olan projeler için idealdir.Ekip içi iletişim yüksektir bu sayede hatalar kolay farkedilip kısa sürede düzeltilir.Kullanıcıdan sürekli geri bildirim istemesi de sorunları hızlı bir şekilde düzeltmeye yöneliktir.Değişien gereksinimlere anlık olarak hızlı bir şekilde tepki verir.Bu gibi avantajlar Scrum’ı güncel ve popüler yapıyor.

HANGİ PROJEDE HANGİ MODEL?

Şimdi ise hangi projelerde hangi modeli kullanıcağımızı konuşalım.Belirsiz öğelerin az olduğu iş tanımlarının net olduğu projelerde V modeli kullanılır.Büyük kitlelere ulaşacak bir proje ise Evrimsel Geliştirme Metodu ‘nu kullanmak fayda sağlayacaktır.Uzun,masraflı ve büyük bir proje için Spiral modeli doğru tercih olucaktır.Küçük veya orta büyüklükteki ve zaman alıcı olmayan projelerde verimli olan Çevik modeller diyebiliriz.Küçük ve gereksinimleri iyi tanımlanmış projelerde Şelale modeli uygun olabilir.

KAYNAKLAR

https://fikirjeneratoru.com/yazilim-proje-yonetimi-yontemleri/

http://furkanalniak.com/yazilim-muhendisligi-yazilim%20/

http://web.firat.edu.tr/mbaykara/ytm2.pdf

http://www.aspmvcnet.com/tr/m/yazilim-muhendisligi/selale-yontemi-waterfall-modeli.html

https://www.quora.com/Why-is-the-Scrum-process-so-popular-in-the-software-industry