İzmir Bakırçay Üniversitesi Bilgisayar Mühendisliği BİL 102 - Ödev 1

Mehmet Ali Öner - 210601043

Giriş

"Yazılım yaşam döngüsünde hangi evreler vardır?", "Bu evrelerin görevleri nelerdir?" gibi sorular ilk kısımda bulunmaktadır. Arkasından metotları\modelleri açıklayacağım. Bu metotları karşılaştırıp, "hangi metot nerede kullanılır?" sorusuna cevap verip scrum'ın neden popüler olduğunu açıklayacağım.

Yazılım Yaşam Döngüsü

Bir yazılım projesi geliştirmek için kodlama bilgisinin yeterli olduğunu düşünüyoruz. Ancak bu yanlıştır. Yazılım projesi sadece koddan ibaret değildir. Yazılımı yazılım yapan geliştirme metotları ve bu metotların doğru seçilip layıkıyla yerine getirilmesidir.

Metotlar yazılım projesinin hatalardan uzak ve verimli bir şekilde nasıl yönetileceğini, yürütüleceğini, ürüne dönüşeceğini belirlemek için ortaya çıkmıştır. Çok sayıda metot mevcuttur.

Metotların hemen hemen her birinde altı evre bulunur. Bu evreler gereksinim/planlama evresi, analiz evresi, tasarım evresi, gerçekleştirme evresi, test evresi, bakım evresi şeklindedir. Bunları birkaç cümleyle anlatmam gerekirse;

Gereksinim/Planlama Evresi: Yazılım projesinin işlevini belirleyecek kullanıcı isteklerinin ve gereksinimlerin belirlendiği evredir. Proje bitene kadar ne kadar maliyeti olacağı, kaç gün süreceği, elde bulunan kaynakların yeterliliği, iş dağılımı gibi planlamaların iş analisti tarafından ya da proje yöneticisi tarafından yapıldığı evredir.

Analiz Evresi: Gereksinim aşamasında toplanan bilgilerin projeye uyumunu ve gerekliliğinin analizinin yapıldığı evredir. Projenin işlevi belirlenir.

Tasarım Evresi: Verileri somutlaştırmaya başladığımız evredir. Proje bu evrede fiziksel olarak tasarlanmaya başlanır. Proje kendi içinde bölümlere ayrılıp bölümler adım adım planlanır ve tasarlanır. Diyagram tasarımları kullanılmalıdır bunun yanı sıra kullanıcı arayüz tasarımı da yapılmalıdır

Gerçekleştirme Evresi: Bu aşamaya başlandıktan sonra herhangi bir tasarım veya analiz yapılmamalıdır. Tasarım aşamasındaki planları ve tasarımları gerçekleştirme koda dökme evresidir. Alfa testleri de (gerçekleştirme evresinin sonuna doğru yapılan ürünü kullanıcılara sunmadan önce olası tüm sorunları gidermek için yapılan testtir.) bu aşamada yapılır.

Test Evresi: Beta testlerinin (gerçek kullanıcılar tarafından gerçekleştirilen, projenin gerçek kullanıcılar tarafından gelecek geri dönüşleri almak için test edildiği aşamadır.) yapılıp geri dönüşlerle projeyi geliştirme evresidir.

Bakım Evresi: Kullanıcıya sunulmuş projede, kullanıcının isteğiyle değişikliklerin, iyileştirmelerin yapıldığı ve hataların giderildiği evredir.

Evreler genel olarak bu şekildedir. Metotlar bu evreleri farklı kombinasyonlarla kullanmaktadır.

Yazılım projesini, yazılım projesi yapan şey metotlardır demiştik bazı projelerde ne metotlar vardır ne de evreler aslında bunlara proje demek pek mantıklı olmaz ama yine makalemde yer vermek istedim. Bahsettiğim projeler gelişigüzel modelle ortaya konmuş projelerdir.

Gelişigüzel Model: Bu modelde belirlenmiş bir yöntem bulunmaz. Basit projeler için kullanılmıştır. Proje tek kişiliktir. Projenin okunması ve bakım yapılması zordur.

Bazı yazılım projelerinde de evreler kullanılır ama metot şeklinde değil herhangi bir geriye dönüş olmadan sırayla kullanılır.

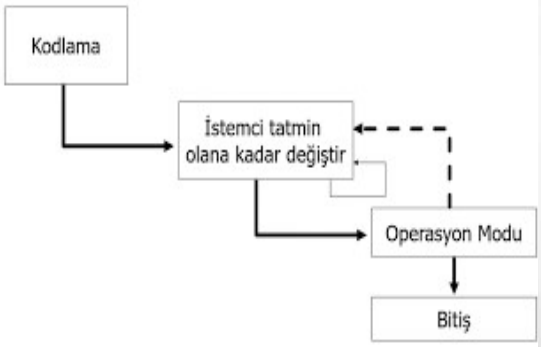
Barok Modeli: Evreler arasında dönüşlerin nasıl yapılacağı tanımlı değildir. Günümüzde kullanılmamaktadır.

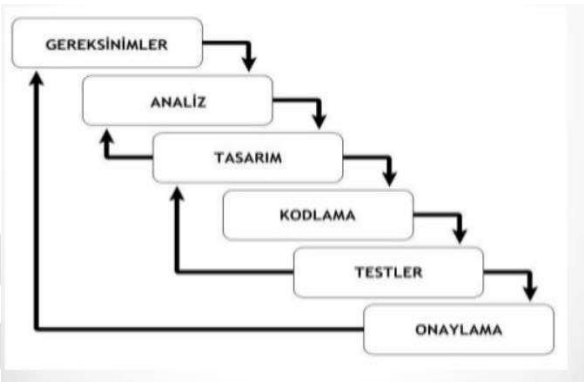
Günümüzde kullanılmayan modeller bunlardı. Günümüzde kullanılan modeller ise küçük projelerde kullanmak için **kodla ve düzelt modeli**, **çağlayan modeli, V modeli** önerilir. Geriye kalan modeller ise **Evrimsel**

Geliştirme Modeli, Prototiplemek, Spiral Model, Artımlı Geliştirme, Yeniden Kullanıma Yönelik Geliştirme, Birleşik süreç, Çevik Modeller, RAD Modeli olmak üzeredir.

Kodla ve Düzelt Modeli: Bu model bir fikirle başlayıp proje bitene kadar kodlamayla devam eder. Kısa ve düşünülmemiş projeler için uygundur. Uzmanlara danışma ihtiyacı yoktur. Kontrol sağlanması zordur.

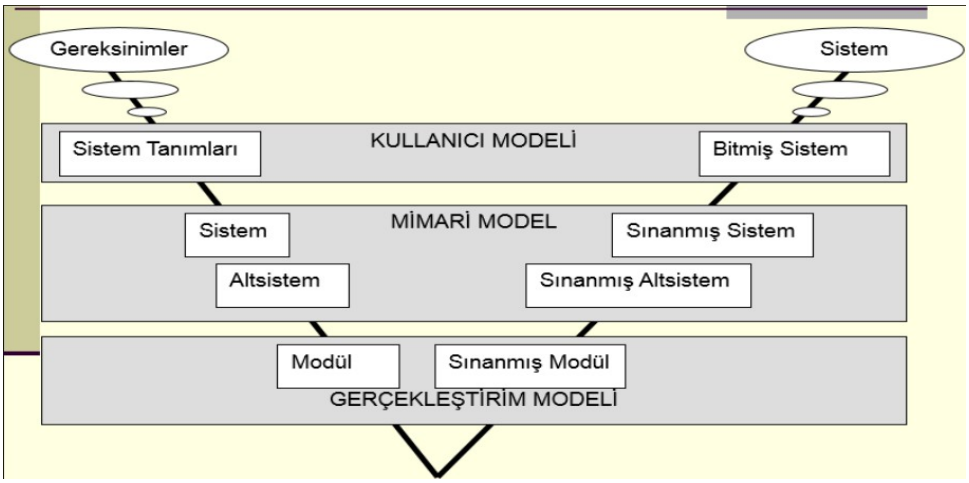
Çağlayan Modeli: En eski ve klasik modeldir. İletişim, planlama, modelleme, inşaat, dağılım olmak üzere 5 adımdır. İletişim gereksinimin müşteriden alındığı evredir. Planlama proje hakkındaki maliyet, zaman, iş gücü gibi hesapların planlandığı evredir. Modelleme tasarımın yapıldığı evredir. İnşaat gerçekleştirmenin yapıldığı evredir. Dağılım testin yapıldığı evredir. Doğru yolda olduğunu olduğunun tespiti için her adımdan sonra geri bildirim alınır. Test projenin bittiği zaman yapıldığı için kodlama riskli olabilir o yüzden küçük projelerde kullanılması gerekmektedir.



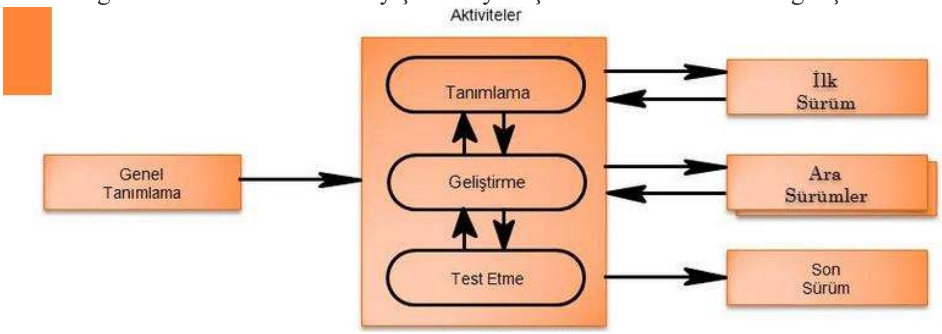


Kodla ve Düzelt Modeli **Çağlayan Model**

V Modeli: Çağlayan modelinin her adımda test edildiği halidir. Kontrol sağlanması kolaydır. Kod ve tasarımsal hatalar kolaylıkla tespit edilebilir. Ancak zaman bakımından maliyetli bir modeldir. Her aşamada test etmek zaman alır.

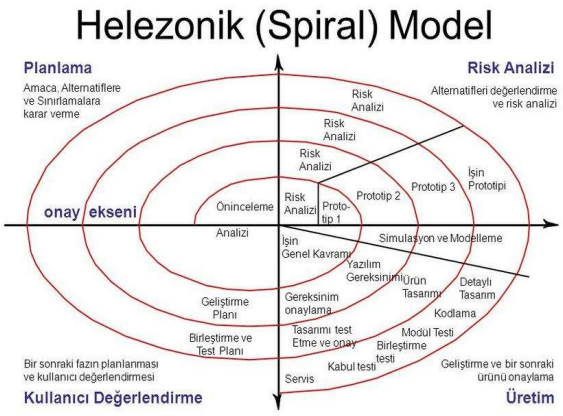


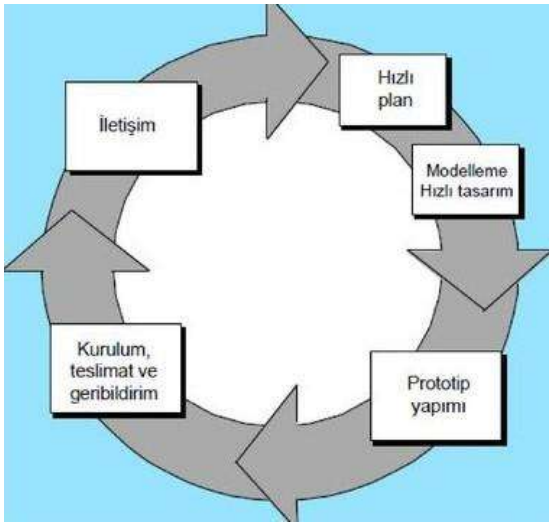
Evrimsel Geliştirme Modeli: Bu modeli "ne istediğimi bilmiyorum ama görsem tanırım" dediğimiz projelerde kullanırız. Diğer modellere oranla ilerleyişi daha yavaştır. İlk evrimin üzerine gelişen bir modeldir.



Prototiplemek: Prototip üretip geliştirme amacında kullanılan modeldir. Gereksinim nu model için çok önemlidir. Hızlı analiz, hızlı tasarım, hızlı kodlama yapma ihtiyacı vardır. Yeni gereksinimlere açıktır. Risk kontrol altındadır.

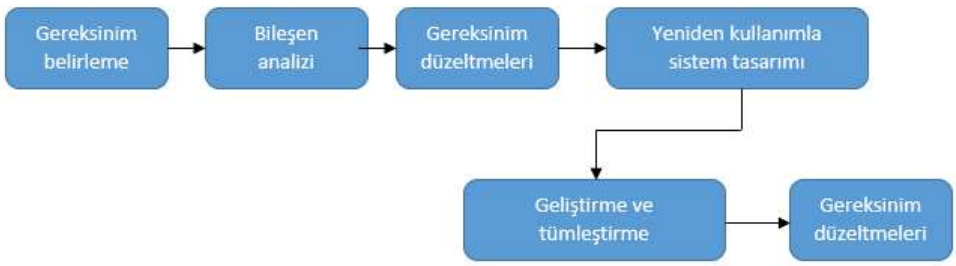
Spiral Model: Modelin amacı risk analizi ve prototip üretmektir. Her döngü bittiğinde risk analizi incelenmiş olur. Her döngü sonunda yeniden planlama yapılır. Yeni prototip üretilir. Kullanıcı projesini önceden görebilir. Hataları erkende giderme şansı verir. Birçok modeli içerisinde barındırır. Eskiden kullanılmış prototipleri kullanmak için uygun bir modeldir. Zaman ve maliyet daha rahat hesaplanabilir. Özellikle güvenlik yazılımlarında kullanılır.



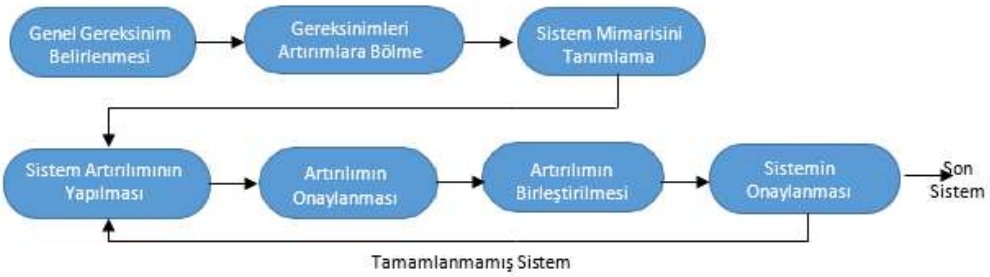


Spiral Model **Prototipleme**

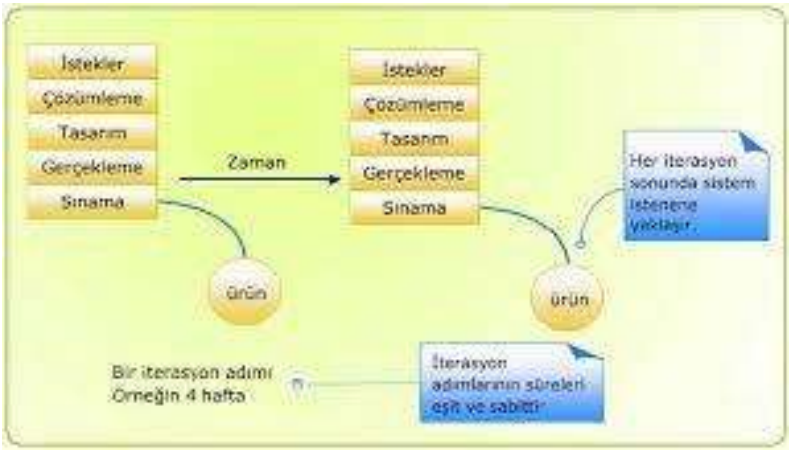
Yeniden Kullanıma Yönelik Geliştirme: Daha önceden hazırlanmış veya dışarıdan temin edilmiş projenin kullanılarak geliştirme yapılmasıdır. Basittir. Yönetimi kolaydır. Kısa sürede yazılım geliştirmesi yapılabilir. Pahalıdır ve uzmanlık gerektirir.

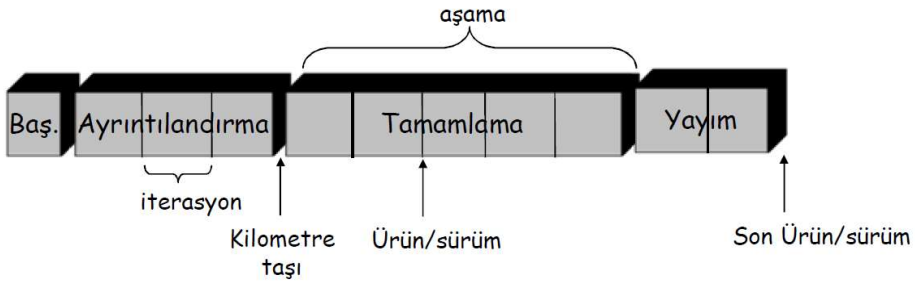


Artımlı Geliştirme: Müşteri ürününde bir değişiklik istiyorsa artımlı geliştirme bu değişikliğe uyum sağlar. Bu model, projeyi adım adım geliştirip teslim etmemizi ister. Kısıtlı sayıda iş gücüyle bu model yürütülebilir. Modelde bir tarafta üretim bir tarafta da kullanım yapılır. Sistem için gerekli ürünler müşterilerle belirlenir. Tüm projenin başarısızlık riski düşüktür. Böl ve yönet yaklaşımındadır.



Birleşik süreç: Nesneye yönelik yazılım geliştirme yöntemlerinin en iyi özelliklerini birleştirilerek oluşturulmuş modeldir. Yinelemeli, arttırmalı ve evrimsel bunlarla beraber risk güdümlüdür. Değişen isteklere kolay uyum sağlar. Ürün erken teslim sağlar. Risk yönetimi zayıftır. Pahalı olabilir.





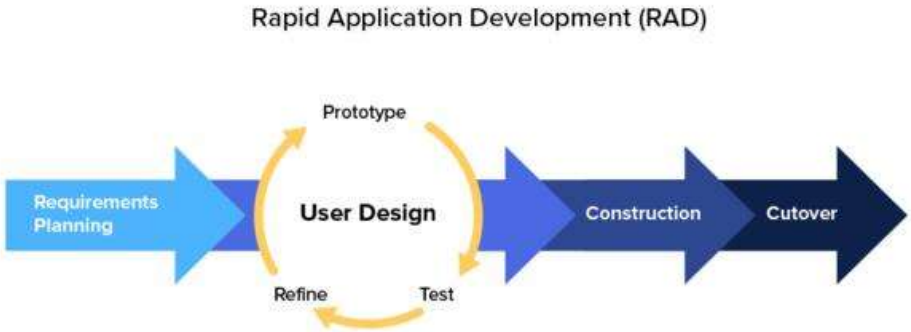
Başlangıç: Vizyon belirleme, fizibilite araştırması, tamam ya da devam kararı.

Ayrıntılandırma: Gerçekçi olarak projenin tamamlanması.

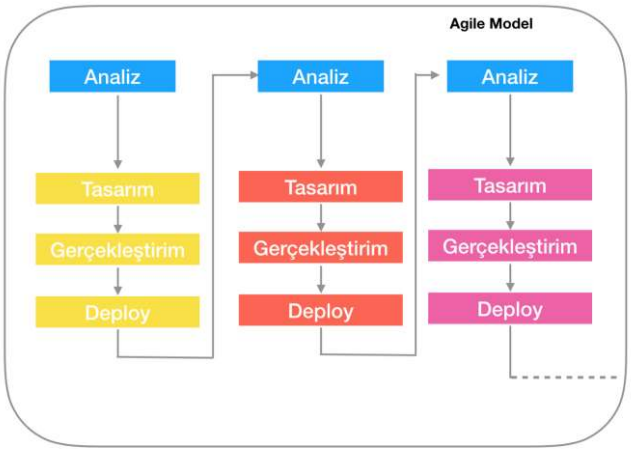
Tamamlama: Daha az riske sahip kısımların yinelemeli olarak gerçekleştirilmesi.

Yayım: Beta testlerinin yapılması ve projenin piyasaya çıkması

RAD Modeli: Hızlı uygulama geliştirme modelidir. Başlangıçta planlama ve ihtiyaçlar belirlenir. Proje küçük parçalara ayrılır, prototipler oluşturulur nu prototipler tamamlandığında birleştirilip teslim edilir. Bu hızlı model beş aşamadan oluşur. İş modelleme ilk aşamadır, planlama ve analiz bu aşamada yapılır. Veri modelleme aşamasında, nesneler nitelik kazanır ve iş modelindeki bilgiler bu nesnelerde kendine yer bulur. Süreç modelleme aşamasında ise veri modellemede tanımlanmış veriler araç haline getirilerek iş modele uygulanmaktadır. Uygulama oluşturma aşamasında ise artık gerçek bir sistem kurulur. Bu aşamada oluşan prototipler bağımsız olarak test aşamasına gider. Her aşamada geri bildirim gerektirdiğinden büyük projelerre göre değildir.



Çevik Modeller: Merkezinde bulunan özellikleri müşteri ile ilişkisi, yinelemeli olması ve erken müşteri geri bildirimidir. Bir kısmı bitirilmiş projenin teslimatı önemlidir. Bu sayede geri bildirimler alıp projeyi müşterinin istediği hale daha çabuk getirilebilir. Testlerden çok müşteri geri bildirimi önemlidir. Ayrıntılı anlatımdan uzak durulmaya çalışılır. Ancak ayrıntılar ve testler olmadığı için sorun tespiti uzun sürmektedir. Bu modelde proje yinelemelere bölünür her yinelemede planlama, ihtiyaçlar, analiz, tasarım, kodlama, birim testi ve kabul testi bulunur. Bu aşamaları doğru atlatan proje parçası müşteriye iletilir. Müşteri geri bildirimiyle müşterinin ihtiyacına uygun proje güncellemeleri yapılır. Eğer müşteri kararlı net değilse bu modelde işler ters gidebilir. Kullanım alanları ise; erken geri bildirim gerektiren başlangıç girişimleri, küçük fonksiyonel parçalara bölünmesi kolay olan projeler, her yineleme aşamasında kademeli olarak geliştirilen projeler. Avantajları arasında bulunan uyumluluğun ve ekibinin moralinin daima yüksek olması çoğu modelde sağlanması oldukça zordur.



Bazı çevik model türleri; Sınırsal programlama (Extreme Programming-XP), Çevik Birleştirilmiş Süreç (Agile Unified Process), Scrum, Test Güdümlü Geliştirme (Test-driven Development), Çevik bilgi Metodu (Agile Data Method), Özellik güdümlü geliştirme (Feature-Driven Programming).

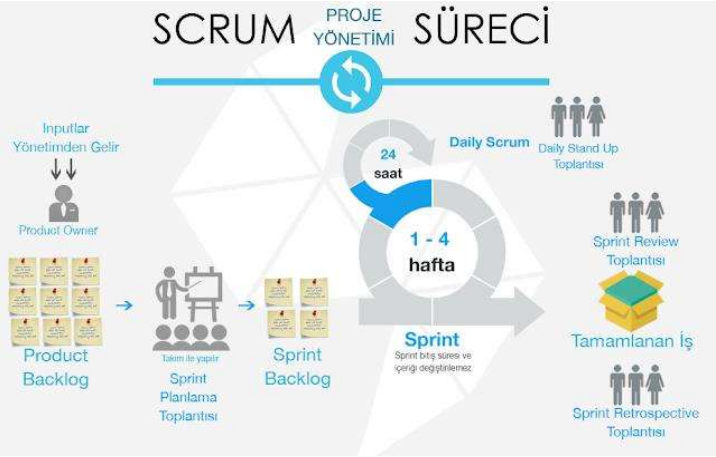
Sınırsal programlama (Extreme Programming-XP): En popüler çevik modellerden birisidir. Müşterinin belirsiz isteklerine ayak uydurabilir. Diğer metotlarda gereksinimler geniş çapta belirlenip, ayrıntılar oldukça fazlaydı XP modelinde ise gereksinimler en az seviyede istenir. Ne kadar az yükle yola çıkılırsa o kadar hızlı gidilir mantığındadır. Bunlardan dolayı büyük tasarımlar, dokümantasyonlar oluşturmaya gerek yoktur.

Scrum: Kompleks yazılım süreçlerinin yönetilmesinde kullanılmaktadır. Bunu yapmak için bütünü parçalayarak, tekrara dayalı bir yöntem izlemektedir. Müşteriden gelecek düzenli geri bildirimlerle ve planlamalarla hedefe ulaşmayı amaçlar. Bu yüzden esnek bir yapısı vardır diyebiliriz. Üç temel prensibe sahiptir. Bunlar; Şeffaflık, denetleme ve uyarlamadır

Şeffaflık; Projenin ilerleyişi, sorunlar herkes tarafından görülebilir olmalıdır.

Denetleme; Projenin ilerleyişi kontrol edilmelidir.

Uyarlamak; Proje yapılan değişikliklere uyum sağlamalıdır.



MODELLERİN KARŞILAŞTIRILMASI

Model/Özellik	Çağlayan Model	Evrimsel Model	Spiral Model	Çevik Model
Gereksinim Belirleme	Başlangıç	Başlangıç	Belirli sıklıkla	Belirli sıklıkla
Maliyet	Maliyetli	Düşük	Maliyetli	Çok yüksek
Başarı Garantisi	Düşük	-	Yüksek	Çok yüksek
Uzmanlık Gerekliliği	Orta	-	Yüksek	Çok yüksek
Fazların Örtüşmesi	Hayır	Hayır	Hayır	Evet
Maliyet Kontrolü	Evet	Evet	Ever	Evet
Basitlik	Basit	Karmaşık	Karmaşık	Karmaşık
Risk Duyarlılığı	Yüksek	Yüksek	Düşük	Azaltılmış
Esneklik	Katı	Düşük	Çok esnek	Çok esnek
Bakım	Düşük	Düşük	Evet	Evet
Değişiklik Yapma	Kolay	Zor	Kolay	Zor
Yeniden Kullanılabilirlik	Düşük	Düşük	Mümkün	Evet
Dökümantasyon ve Eğitim Gerekliliği	Evet	Evet	Evet ama çok değil	Evet
Zaman	Çok Uzun	Uzun	Uzun	Kısa
Uygulama	Kolay	Kolay	Karmaşık	Kolay

Hangi Projede Hangi Modeli Kullanmalıyız ?

Bu sorunun cevabını olay örgüsü şeklinde anlatmak istiyorum.

Ne yapacağını bilmeyen, aklında sadece küçük bir fikirle proje oluşturmak isteyen kişiler **Kodla ve Düzelt Modeli** ile güzel sonuç elde edebilir.

Ne yapacağını bilen ve küçük bir proje geliştirmek isteyen kişi ise **V Modeli** tercih edebilir.

Düşük maliyet ve zamanın bol olduğu projelerde **Evrimsel Geliştirme Modeli** kullanılabilir.

Proje fikri bir kalıba oturtulmamış, düşük maliyetli, esneklik bakımından rahatlık isteyen projelerde **Prototipleme** modeli kullanılabilir.

Proje fikri değişebilecek, esneklik isteyen ve başarı yüzdesi yüksek olması istenilen projelerde **Spiral Model** kullanılabilir.

Yüksek garanti istenilen, prototipler halinde teslim edilen, esnekliğin fazla olduğu projelerde **Birleşik Modeller** kullanılabilir.

Daha önceden hazırlanmış veya dışarıdan temin edilmiş projenin kullanılarak geliştirme yapılmasına izin verilmesinin mümkün olduğu model **yeniden kullanıma yönelik geliştirme modelidir.**

Müşteri ürününde bir değişiklik istiyorsa **artımlı geliştirme modeli** bu istek için kullanılacak en güzel modellerdendir.

Çok fazla detaya girilmeden, hızlı şekilde çalışabilen bir uygulama geliştirmek istenilen projelerde **RAD Modeli** kullanılabilir.

Ayrıntıların istenmediği, hızlı hızlı prototipler üretip müşteriye sunulup, müşterinin geri dönüşleriyle projenin şekillenilmesinin istenildiği projelerde **Sınırsal Programlama(XP) Modeli** kullanılabilir.

Anlaşılmasının zor, prototipler halinde müşteriye teslimi istenilen, başarısı yüksek olması istenilen projelerde **Scrum Modeli** kullanılabilir.

Scrum günümüzde neden popüler?

Scrum çoğu şirketin kullandığı bir modeldir. Başarı oranının çok yüksek olması bu modeli göz önüne çıkarmaktadır. Bu başarısının sebebi ise karmaşık projeleri basite indirgeyerek yani parçalayarak küçük küçük işler haline getirmesidir. Yinelemeler şeklinde bu parçalar ürüne dönüştürülüp (her yineleme en fazla 30 gün sürebilir, istisnalar hariç) . Müşteri teslim edilmiş kısmı inceleyip geri dönüşler yapar. Bu geri dönüşle yeni planlamalar yapılır. Yeni bir yineleme başlar...

Scrum'ın kolay uyum sağlaması da onu göz önüne çıkarmaktadır.

https://medium.com/@omerharuncetin/yazılım-yaşam-döngü-modelleri-543c7879a742

https://osmanozaydin.com/yazilim-yasam-dongusu-ve-agile-yazilim-gelistirme/

https://tr.linkedin.com/pulse/yazılım-yaşam-döngüsü-nedir-veysel-ugur-kizmaz

İzmir Bakırçay Üniversitesi - Hafta 3 / Hafta 2 Yazılım Yaşam Döngüsü Modelleri Deniz KILINÇ

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

https://enprobilisim.com/yazilim-gelistirme-sureci-modelleri-sdmp/

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

https://medium.com/@ahmetuyar/extreme-programming-xp-nedir-ddc003a515c4

https://e-bergi.com/y/cevik-modelleme-ve-cevik-yazilim-gelistirme/

https://medium.com/@secilcor/scrum-nedir-6a4326951dd8