İZMİR BAKIRÇAY ÜNİVERSİTESİ

MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİL102 – YAZILIM MÜHENDİSLİĞİ TEMELLERİ

Ödev 1

Yazılım Yaşam Döngü Modelleri

Seda Yaprak 190604003

İzmir, 2023

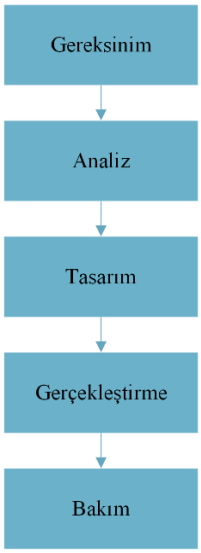
Dr. Öğr. Zekeriya Anıl GÜVEN

İÇİNDEKİLER

1. GİRİŞ

Döngü yaklaşım olan yazılım yaşam döngüsü, bilgisayar tabanlı sistemi ya da alt sistem oluşturma veya bakımında izlenen ve geliştirilen süreçtir. Yazılım yaşam döngüsü temel olarak planlama, tanımlama, tasarım, geliştirme, entegrasyon ve test, uygulama ve bakım aşamalarından oluşmaktadır (Şeker, 2015).

Yazılım yaşam döngüsüne ait şema şekil 1'de verilmiştir.



Şekil 1. Yazılım yaşam döngüsü aşamaları

Döngünün aşamaları temelde aynı olsa da aşama ve adlandırma konusunda fikir farklılıkları olabilmektedir. Fikirler incelendiğinde tüm açıklamaların aynı ana fikirde olduğu gözlemlenir (Ersoy, 2002).

Başlangıç aşaması olduğu için en önemli aşama olan plan safhası, projenin oluşturulduğu, fikirlerin ortaya atıldığı ve tartışıldığı aşamadır. Analiz safhası, fikrin ne olduğunun tartışıldığı aşamadır. Tasarım safhasında sistem ya da yazılım tasarımları yapılarak karar verilir, ekran geçişleri, fonksiyonel adımlar, yazılım bileşenleri, modül gibi tasarımlar yapılır. Gerçekleştirme safhası, tasarım safhasındaki kararlar sonucunda projenin gerçekleştirimi yapılır. Son olarak bakım safhasında yeni tanımlamalar, hatalar meydana gelir ve yeni tasarımlar yapılır. Bu döngü mühendislik projesi olan yazılımın yaşam süresinin sonuna kadar devam eder (Şeker, 2015).

2. YAZILIM DÖNGÜ MODELLERİ

2.1 Şelale (Waterfall) Modeli

Statik bir model olan Şelale modeli, basit ve diğer modellerin kaynağı olması sebebiyle önemlidir. İlk aşaması planlamadır sonrasında durum ve istek analizi olarak ikiye ayrılan analizler yapılır. Durum analizi eldeki kaynaklar, geliştirme ortamı gibi durumların analizini yapar. İstek analizi ise istenilen nedir sorusu ile ilgili analiz yapar. Bir sonraki aşama olan tasarım aşamasında ise analiz safhasındaki çıktılara bakılarak geçilir ve fonksiyonel özelliklerin tasarımı, ekran tasarımı gibi tasarımlar yapılır. Nesne tabanlı ortam kullanılacak ise tasarım aşaması class diyagramları, use case çizimi gibi süreçler sebebiyle tasarım aşaması uzayabilmektedir. Tasarım aşamasından sonraki aşama uygulama aşamasıdır ve test, tasarım yanlışları, analiz yanlışları gibi alt aşamalara ayrılabilir. Uygulama aşaması kodlama aşaması olarak da düşünülebilir. En son aşama ise sistemin genel testi ve müşteriye teslim edilmesidir (Şeker, 2015).

Planlama ile test aşamasının da dahil olduğu uygulama aşamasına kadar sistemde değişiklik yapma şansı yoktur. Yeni bir şey yapılacaksa planlama tekrarlanır ve yeni proje olarak değerlendirilir. Bu yüzden de uzun süreç gerektiren örneğin 1 ya da 2 yıl gerektiren bilişim projelerinde aşamaların tamamlanması beklendiği ve müdahale etme şansı olmadığı için tehlikelidir (Şeker, 2015).

Bu modelde bir önceki aşamaya dönüş sağlanabilmektedir. Örneğin tasarım safhasında problem olduğunda analiz safhasına, uygulama aşamasında sorun olduğunda tasarım safhasına dönüş yapılabilmektedir. Modelin ilk aşamalarında hata maliyeti daha az iken aşamalar ilerledikçe hata maliyetleri giderek artmaktadır. Örneğin analiz aşamasında bir hatanın fark edilip düzeltilmesi daha az maliyetliyken test aşamasında hata fark edilip analiz aşamasına dönüldüğünde arada geçen süreç sıfırdan oluşturulacağı için maliyet çok artar (Şeker, 2015)

2.2 Yap ve Düzelt (Build & Fix) Modeli

Modelsizlik olarak tanımlanan bu modelde yazılım, ihtiyaçlar doğrultusunda hazırlanır ve müşteri memnuniyeti sağlanana kadar devam eder. Bu modelde analiz ve tasarım aşaması bulunmaz ama müşteri memnuniyeti sağlanana kadar devam ettiği için çok uzun bir geliştirme döngüsüne sahiptir. Bu uzun süreç ve sürecin oluşturduğu yüksek maliyet nedeniyle bu model tercih edilmez. Küçük boyutlu yazılım ihtiyaçlarında uygulama basitliği sebebiyle tercih edilebilmektedir (Ersoy, 2002).

2.3 Sarma (Spiral) Model

Yazılım yaşam döngüsü ve prototip modellerinin en iyi özelliklerini birleştiren bu model, risk analizini de barındırır. Spiralin etrafındaki ilk çevrede hedef, alternatif ve sınırlama tanımlanır, risk analiz edilir. Aşamaları ise şu sırayı takip eder:

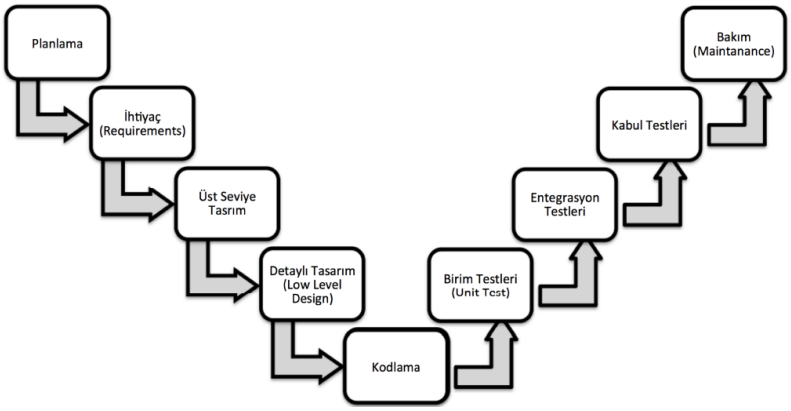
1. Planlama : hedef, alternatif, sınır belirlenir.
2. Risk Analizi: alternatif ve risk tanımının analizi yapılır.
3. Mühendislik: bir düzey sonraki ürün geliştirilir.
4. Müşteri Değerlendirmesi: mühendislik sonuçları değerlendirilir.

Analizde bir sorun varsa mühendislik aşamasında prototip oluşturulur ve geliştiren ile müşteriye yardımcı olur. Bu model sonraki problem tanımlama ve gereksinim iyileştirilmesinde kullanılır (Ersoy, 2002).

2.4 V Süreç Modeli

Aşamaları V şeklini oluşturduğu için bu adı alan model, şelale modeline benzemekle birlikte tasarım yapıldıktan sonra alt seviye tasarımlar yapıldığı için farklıdır. Aşağı doğru bir akışa sahiptir, sonrasında aşamalar yukarı doğru çıkmaktadır (Şeker, 2015).

Modele ait gösterim şekil 2'de görülmektedir.



Şekil 2. V süreç modeli

Bu yaklaşımın özelliği şekli itibariyle aynı hizada olan safhalar birbirini takip eder. Örneğin tasarım yapıldıktan sonra birim testleri gibi testlerin yapıldığı test aşaması gerçekleştirilir. Üst seviye tasarımdan sonra karşısında olan modüllerin birbiri ile uyumlu çalışıp çalışmadığını test eden entegrasyon testleri yapılır. Analiz aşamasında müşteri ihtiyaçları belirlendikten sonra kabul testleri yapılır (Şeker, 2015).

2.5 Scrum

Bir sistem ya da yeni ürün prototipi için yönetim, gelişim ve bakım süreci olan Scrum yüksek kaliteli yazılımın düzenli ve sık teslimi süresince müşteriye iş değerini doğru ve önce teslimine izin veren iteratif ve nesne tabanlı geliştirme sunan metodolojidir. Scrum piyasada kabul görmüş, kullanılan, pratiklere ve tecrübeye dayanan süreç teorisi sonucu ortaya çıkmıştır (Baytam, 2011).

Scrum, ExtremeProgramming(XP), Feature-Driven Development (FDD), Test Driven Development (TDD), Agile, Dynamic System Development Methodology (DSDM), UnifiedProcess, Microsoft Solution Framework (MSF) ve LEAN Development gibi yaklaşımları benimser. Scrum ile karmaşık yazılım ve ürünler adım adım geliştirilerek kontrol ve yönetim sağlanır. Scrum'ın sağladığı avantajların başında gereksinimleri sağlayan, başarılı ve kaliteli bir yazılım için birlikte olan takım ve müşteriyi sağlaması gelir (Baytam, 2011).

Projelerde az kişiden oluşan çok sayıda ekip vardır, ekip ve iş birimleri arasında iş birliği vardır. Ekibin ilerlemesi risk ve karmaşıklık göz önüne alınarak 1 ila 4 hafta içerisinde devamlı incelenir, bu kontrol mekanizması sayesinde öngörülemeyen durumların yönetilmesi ve riskin kontrol altında tutulması sağlanır. Bunun sonucu olarak duyarlılık, güvenilirlik ve esneklik kazanılır.Yani Scrum metodolojisinde üretim esnektir yani içerik çevre tarafından oluşturulur, proje başlangıcına göre erken veya geç tamamlanabilir (Baytam, 2011).

Scrum, nesne tabanlıdır yani her takım, davranış ve arayüzden oluşan nesneler ile çalışır. Bu yaklaşımla birlikte rekabet unsuru göz önüne alındığında yazılım şirketlerinin başarı yakalamasını sağlar (Baytam, 2011).

Scrum'ın sağladığı bir diğer fayda ise geri bildirimle ile kısa dönemli planlamalarla hedefe ulaşmaktır. Geliştirme boyunca önem ve ihtiyaç sırasını belirleyerek çalıştığı için büyük parçalar yerine önce oturtulur, müşteri ihtiyaçları doğrultusunda parçalar şekillenir. Ayrıca Scrum, belirsizliğin fazla olduğu projelerde bu sebeple günümüzde yoğun biçimde uygulanmakta ve başarılı sonuçlar üretmektedir (Baytam, 2011).

2.6 Modellerin Karşılaştırılması

Farklı ihtiyaçlara çözüm üreten Şelale, Spiral, SCRUM, Yap ve Düzelt modellerine ait karşılaştırma tablo 1'de verilmiştir.

Tablo 1. Yazılım yaşam döngü modellerinin karşılaştırılması

Özellik	**Şelale Modeli**	**Yap ve Düzelt Modeli**	**Spiral Model**	**SCRUM Modeli**
Proje Maliyeti	Maliyetli	Düşük	Değişken	Çok yüksek
Başarı Olasılığı	Düşük	Düşük	Orta Düşük	Yüksek
Uzman Gerekliliği	Orta	Az	Gerekli	Çok yüksek
Risk Duyarlılığı	Yüksek	Yüksek		Az
Bakım	Düşük	Düşük		Gerekli
Dokümantasyon	Gerekli	Gerekli	Gerekli	Gerekli
Esneklik	Esnek değil	Esnek değil	Esnek değil	Çok esnek
Zaman	Uzun süreç	Uzun süreç	Değişken	Kısa süreç
Proje Büyüklüğü		Küçük		Büyük

3. SONUÇ

Bu çalışmada yazılım proje yönetiminde önemli olan yazılım yaşam döngüleri incelenmiş, modellerin açıklaması, karşılaştırılması ve hangi projeler için daha uygun oldukları anlatılmıştır. Çalışma sonucunda müşterinin ihtiyacına ve gereksinimlerine göre farklı modellerin kullanılabileceği, her modelin kendisine göre avantajları ve dezavantajları olduğu görülmüştür. Müşterinin ihtiyaçları, memnuniyeti, maliyet, zaman gibi kısıtlar çerçevesinde ihtiyaca uygun model belirlenmeli ve yazılım yaşam döngüsü başlamalıdır. Bu süreç yazılımın ömrünü tamamlayana kadar devam etmelidir.

KAYNAKÇA

Baytam, V. (2011). *Scrum Yazılım Geliştirme Metodoloji İçin Yönetim Sistemi Tasarımı ve Gerçeklenmesi*.

Ersoy, E. (2002). *Yaşam Döngüsü Yönetimi Kavramının Yazılım Ürünlerine Uygulanması*.

Şeker, S. E. (2015). Yazılım Geliştirme Modelleri ve Sistem/Yazılım Yaşam Döngüsü. *YBS Ansiklopedi*.

İLETİŞİM

LinkedIn: https://www.linkedin.com/in/seda-yaprak/

Github: https://github.com/YSeda

Medium: https://medium.com/@seda.35yaprak