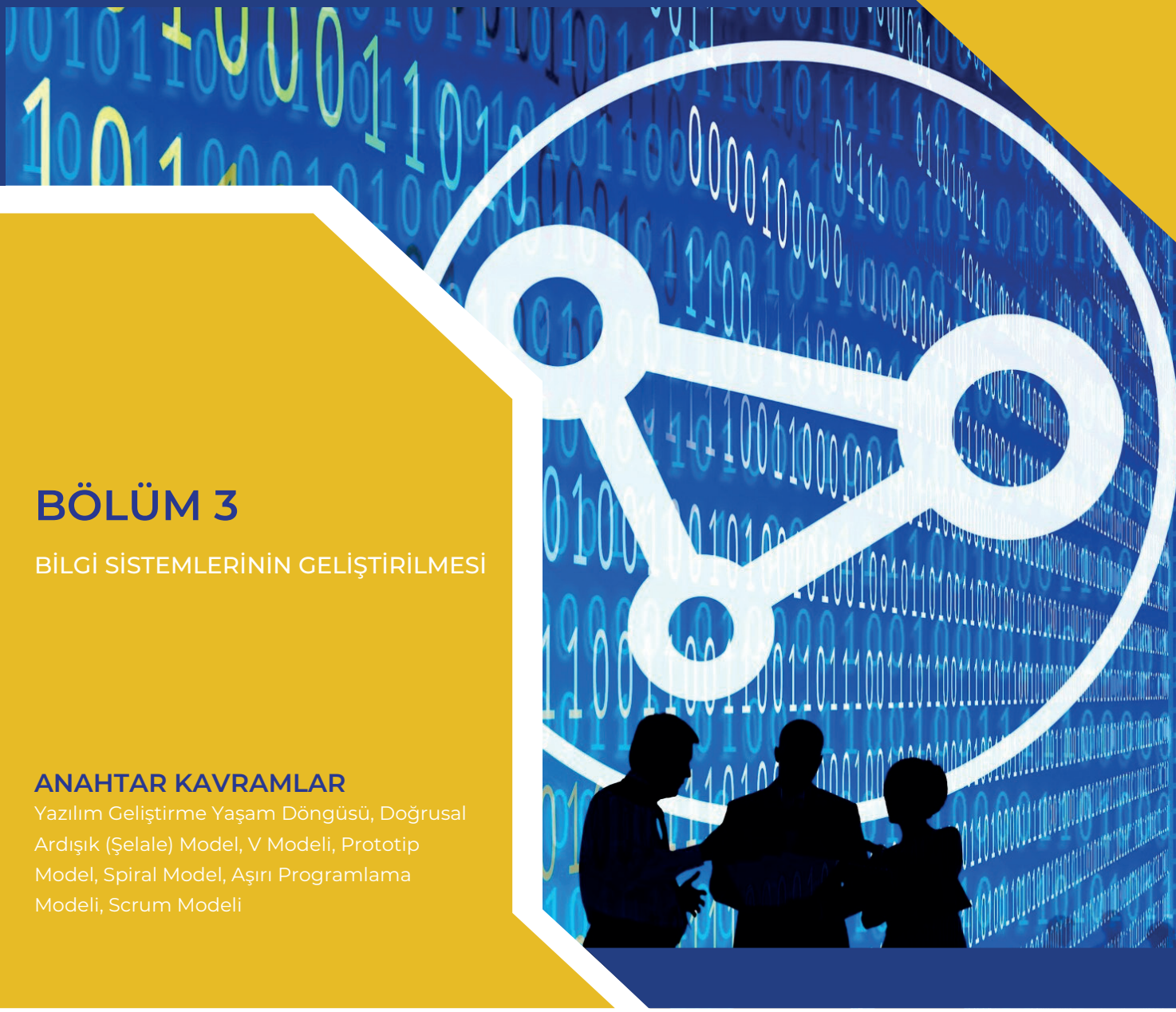


BÖLÜM 3

BİLGİ SİSTEMLERİNİN GELİŞTİRİLMESİ

ANAHTAR KAVRAMLAR

Yazılım Geliştirme Yaşam Döngüsü, Doğrusal Ardışık (Şelale) Model, V Modeli, Prototip Model, Spiral Model, Aşırı Programlama Modeli, Scrum Modeli

ÖĞRENME HEDEFLERİ

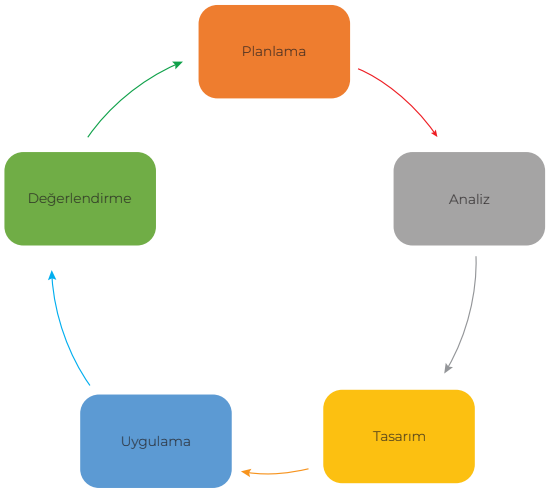
1. Sistematik yazılım geliştirme hakkında bilgi sahibi olacaksınız
2. Yazılım Geliştirme Yaşam Döngüsünü ve aşamalarını açıklayabileceksiniz
3. Yazılım Geliştirme Modelleri hakkında bilgi sahibi olacaksınız
4. Geleneksel ve Çevik yazılım geliştirme modelleri hakkında bilgi sahibi olacaksınız

ÜÇÜNCÜ BÖLÜM

GİRİŞ

Günümüzde bilgi sistemleri eğitim, sağlık, finans, seyahat ve eğlence gibi yaşantımızın neredeyse tüm alanlarında kullandığımız sistemlerdir. Bilgi sistemleri sayesinde birçok iş ve işlem zahmetsiz ve hızlı bir şekilde tamamlanabilmektedir. Davis (1999, s. 3) bilgi sistemini, doğru zamanda, doğru kişiye, doğru veri ve bilgiyi sağlamayı amaçlayan donanım, yazılım, veri, insan ve prosedürel bileşenler kümesi olarak tanımlamaktadır. Tüm sistemlerin geliştirilme sürecinde olduğu gibi bilgi sistemlerinin geliştirilmesinde de birçok farklı model rehber alınmaktadır. Bu bölümde bilgi sistemlerinin geliştirilmesinde kullanılan modeller hakkında bilgi verilecektir.

Sarıdoğan (2008, s. 63) her sistemin uygun bir yaşam döngüsü (Life Cycle) metodu kullanılarak proje şeklinde gerçekleştirilmesinin etkinlik açısından çok önemli olduğu vurgulamaktadır. Sistematik olarak geliştirilen projelerin geçirdiği süreçleri açıklayan birçok modelin ortak noktalarını tanımlamak için sıkça "*Sistem Geliştirme Yaşam Döngüsü (System Development Life Cycle-SDLC)*" kavramının kullanıldığı görülmektedir. Bu ortak model temel olarak planlama, analiz, tasarım, uygulama ve değerlendirme adımlarını içermekte ve bir döngü şeklinde uygulanmaktadır. SDLC Şekil 1' de görsel olarak sunulmuştur.



Şekil 1- Sistem Geliştirme Yaşam Döngüsü (SDLC)

1. SİSTEM GELİŞTİRME YAŞAM DÖNGÜSÜ

1. Adım: Planlama

Planlama SDLC' nin ilk adımıdır. Bu adımda; projeye neden ihtiyaç duyulduğu, ne kadar zamanda tamamlanması gerektiği, ne kadar bütçe gerektiği ve projenin kazanımlarının neler olacağı gibi soruların yanıtlanması gerekmektedir. Planlama adımının en kritik yanının projeye devam edip edilmeyeceğine dair kararın çoğunlukla bu aşamanın sonunda verilmesi olduğu düşünülmektedir. Bu adımda alınacak kararla projeden vazgeçilebilir, uygulanabilir veya genişletilerek uygulanabilir. Silahtaroğlu (2010, s. 31) bu adımda yapılacak işlemleri beş başlıkta toplamaktadır. Bunlar:

1. Problemin tanımlanması,

2. Fizibilite raporlarının hazırlanması,

3. Proje zaman çizelgelerinin hazırlanması,

4. Projede çalışacak personelin zamana bağlı olarak belirlenmesi ve

5. Projenin başlatılmasıdır.

Planlama adımının doğru olarak ele alınması projenin başarıya ulaşması açısından çok önemlidir. Çünkü hatalı hesapla başlayan bir işin doğru sonuçlandırılması olası olmadığı düşünülmektedir. Bu adımın sonunda yapılan işlemler raporlanır ve projeye devam kararı verilirse ikinci adıma geçilir.

2. Adım: Analiz

Analiz adımında projenin ihtiyaçları belirlenmeye çalışılmaktadır. Sistemin girdi ve çıktılarının neler olacağı, kimler tarafından kullanılacağı, kullanıcıların beklentilerinin neler olduğu gibi soruların cevaplanması gerekmektedir. Bu tip soruları cevaplamak için de bilgi toplanması, tanımlar yapılması, örnek senaryolar hazırlanması, alternatif çözüm önerilerinin getirilmesi ve tüm bu verilerin değerlendirilmesi gerekmektedir. Bu adımda da yapılan tüm işlemler raporlanarak kayıt altına alınmaktadır. Foster (2022, ss. 90-91) yazılımsal bir sistem geliştirilirken bu adımda yapılması gerekenleri şu şekilde sıralamaktadır:

a) Katkı sağlayacak kişi veya kurumların tanınması,

b) Problemin özetlenmesi,

c) Sistemin genel durumunun çıkarılması,

d) Sistem bileşenlerinin belirlenmesi,

e) Gereksinimlerin detaylı olarak belirtilmesi,

f) Verilerin nasıl depolanacağının belirlenmesi,

g) İşletim için gereksinimlerin belirtilmesi ve

h) Sistemin kurallarının tanımlanması.

3. Adım: Tasarım

Bu adım, analizde elde edilen veriler temelinde geliştirilecek sistemin tasarlandığı adımdır. Bu adımda sistemin nasıl görüneceği ve nasıl çalışacağına dair eskizler hazırlanmaktadır. Kullanıcı ve sistem arayüzünün nasıl olacağı, bileşenlerin birbiriyle nasıl etkileşime gireceği, veritabanı ve sistemle nasıl entegre olacağı, sistem kontrollerinin nasıl yapılacağı gibi başlıklar tasarlanıp görselleştirilmektedir.

Yapılan tasarımda en çok üzerinde durulan konuların, sistemin tutarlı ve bütünlüğe sahip olması olduğu belirtilmektedir (Tsui vd., 2017, s. 170). Tutarlı bir sistem tasarımı ekranlarda, raporlarda, veritabanı öğelerinde ve süreç mantığında ortak terminolojinin kullanılmasını sağlamaktadır. Bütünlüğe sahip bir tasarım ise tüm gereksinimlerin tasarlandığı ve hiçbirinin dışarıda bırakılmadığı tam bir yapı sunmaktadır.

4. Adım: Uygulama

Uygulama adımında şekillenen sistemi kurmak hedeflenmekte ve tasarım adımında belir-

lenen arayüzler, ilişkiler, veritabanı ve kontroller gerçekleştirilmektedir. Bu adımda sistem belirlenen yazılım dilinde kodlanır ve veritabanı yönetim sistemleri işe koşulur. Uygulama adımının başarıya ulaşması için geliştirilen sistemin şu özellikleri taşıması gerektiği belirtilmektedir (Tsui vd., 2017, s. 192):

a) Okunabilirlik: Kodlar başka programcılar tarafından da anlaşılabilmelidir.

b) Sürdürülebilirlik: Kodlar kolayca değiştirilebilmeli ve bakımı yapılabilmelidir.

c) Performans: Diğer her şey sabitken mümkün olduğu kadar hızlı çalışan kod üretmelidir.

d) İzlenebilirlik: Tüm kod öğeleri bir tasarım öğesine karşılık gelmelidir. Kod, tasarıma kadar izlenebilir olmalıdır.

e) Doğruluk: Sistem, analiz ve tasarım adımlarında amaçlanan şeyi yapmalıdır.

f) Bütünlük: Tüm sistem gereksinimleri karşılanmalıdır.

5. Adım: Değerlendirme

Bu adım bakım (Silahtaroğlu, 2010), destek (Sarıdoğan, 2008), entegrasyon, test ve değerlendirme (Wasson, 2006) gibi farklı isimlerle de anılmaktadır. Değerlendirme adımının amacı ortaya konulan sistemin doğru çalıştığından emin olmak ve varsa eksik ya da hatalı yönlerinin tespit edilerek düzeltilmesini sağlamaktır. Bu adımda elde edilen veriler sistemin yeniden ele alınması için birinci adımdan itibaren girdi olarak kullanılabilmektedir. Bunun yanında başka adımlara da doğrudan veri sağlayıp, oradaki eksikliklerin giderilmesinde de kullanılabilmektedir. Bu adımda yapılan işlemler şu şekilde özetlenebilir:

a) Kullanıcı desteği verilmesi,

b) Hataların giderilmesi,

c) Versiyon veya sürüm yükseltmelerinin yapılması,

d) Mevzuat değişikliklerinin sisteme entegre edilmesi,

e) Teknoloji değişikliklerinin yapılması.

2. YAZILIM GELİŞTİRME MODELLERİ

Sistem Geliştirme Yaşam Döngüsünde bahsedilen adımların özellikle yazılım sistemi geliştirmek için tasarlanmış birçok farklı model bulunmaktadır. Bu modellerin amacı yazılım sistemi geliştirilirken zaman, para, işgücü gibi kaynakların doğru kullanımını sağlamak ve elde edilen ürününün hem üretici hem de müşteri için tatmin edici olmasını garanti etmektir. Bu başlık altında yazılım geliştirme modelleri iki kategoride ele alınmakta ve yaygın kullanılan modeller tanıtılmaktadır.

A. Geleneksel Yazılım Geliştirme Modelleri

B. Çevik (Agile) Yazılım Geliştirme Modelleri.

A. Geleneksel Yazılım Geliştirme Modelleri

Bu kategoride yer alan yazılım geliştirme modellerinin, çoğunlukla işletmeler için kullanılan sistem geliştirme adımlarının, yazılım geliştirme adımlarına uyarlanması sonucu ortaya çıktığı görülmektedir. Bu sebeple kategori adı Geleneksel Yazılım Geliştirme Modelleri olarak seçilmiştir. Bu başlıkta tanıtılacak modeller şunlardır:

1. Doğrusal Ardışık (Şelale) Model

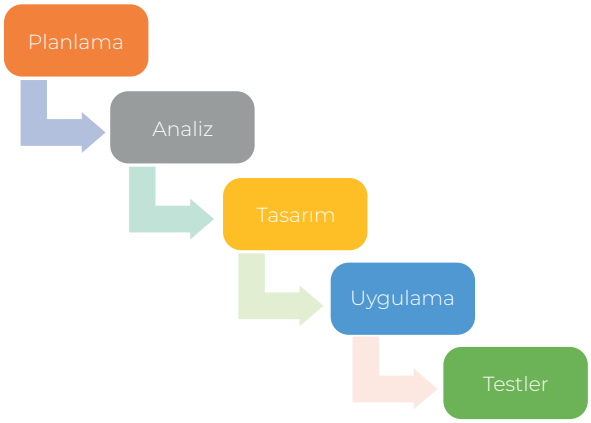
2. V Modeli

3. Prototip Model

4. Spiral Model

1. Doğrusal Ardışık (Şelale) Model

Doğrusal Ardışık Model (DAM) katı bir dokümantasyon ve sürece sahiptir. Her aşaması sıra ile takip edilir, biri tamamlanmadan diğerine geçilmez ve biri tamamlandıktan sonra önceki aşamalara dönülmez. Bu sebeple hedefleri çok net bir şekilde tanımlanmış ve süreç içerisinde değişiklik beklenmeyen projeler için uygulanabilirdir. DAM beş aşamadan oluşmaktadır ve bunlar Şekil 2'de gösterilmiştir.



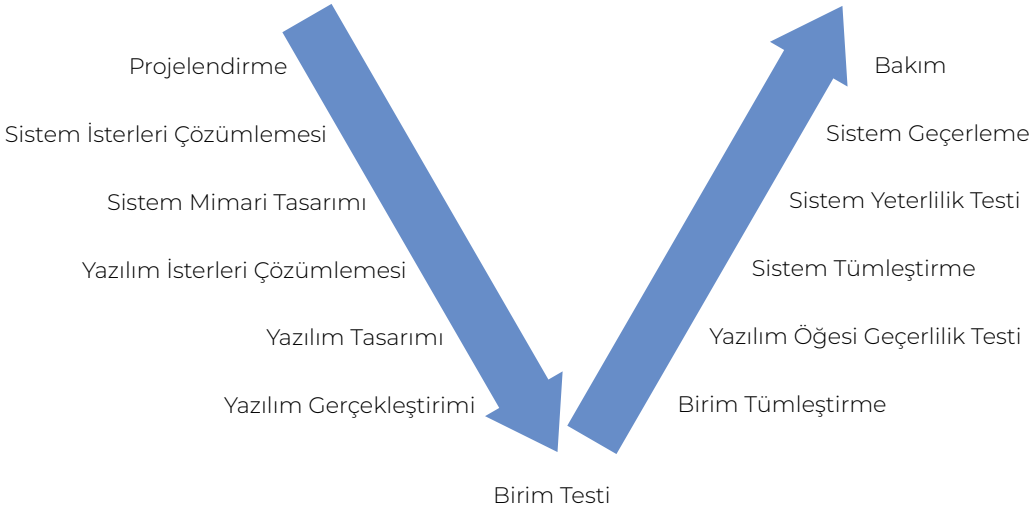
Şekil 2- Doğrusal Ardışık (Şelale) Model

DAM Planlama aşaması ile başlar. Bu aşamada projenin bütçesi, süresi, ve paydaşları gibi hususlar belirlenir ve Analiz aşamasına geçilir. Analiz aşaması genel olarak durum ve ihtiyaç analizi olmak üzere ikiye ayrılır. Durum analizinde var olan kaynaklar ortaya konur ve yazılım geliştirme ortamına karar verilir, ihtiyaç analizinde ise projeden ne beklendiği açıklanır ve bu bilgilere göre Tasarım aşamasına başlanır. Tasarım aşaması en detaylı çalışmanın yapıldığı aşamadır. Bu aşamada yazılım ekranları, akış diyagramları, veri tabanı gibi yazılım bileşenlerinin detaylı tasarımları yapılır. Uygulama aşaması, tasarım aşamasında karar verilen tasarımların kodlara döküldüğü aşamadır. Yazılımın kendisi önceki aşamalarda verilen kararlar doğrultusunda kodlanır. Ardından son aşama olan Testler aşamasında elde edilen yazılımın varsa hataları tespit edilerek giderilir ve proje tamamlanır. Her aşamada yapılan işlemler detaylı olarak yazılır ve rapor haline getirilir. (Şeker, 2015)

2. V Modeli

V modeli süreç gösteriminden dolayı bu ismi almaktadır. V şeklinin sol tarafı üretimi, sağ tarafı ise test işlemlerini göstermek için kullanılmaktadır. Modelin aşamaları Şekil 3'te gösterilmektedir.

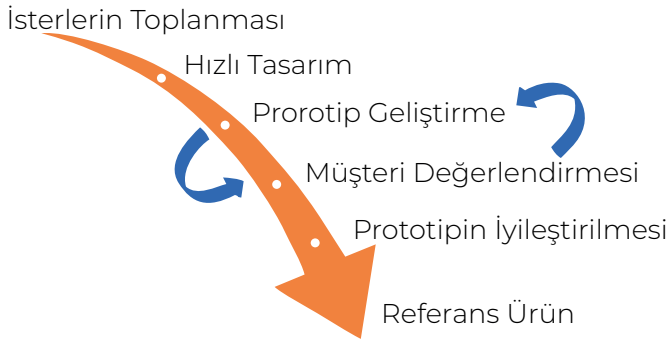
V Modelinde sağ tarafta yer alan süreç, DAM başlığında değindiğimiz Planlama, Analiz ve Tasarım aşamalarında yapılan işlemlerin, yazılım projelerine özel olarak isimlendirilmiş aşamalarından oluşmaktadır. Model görselinin sol tarafı ise test aşamalarından oluşmaktadır. Son aşama olan Bakım ise test aşamalarında elde edilen veriler ışığında yazılım projesinde yapılması gereken iş ve işlemlerin yapıldığı aşamadır.



Şekil 3- V Modeli

3. Prototip Model

Prototip Model, proje paydaşlarının bir araya gelerek yazılımın girdi ve çıktılarına, verilerin nasıl yönetileceğine ve korunacağına karar vermesiyle başlamaktadır. Bu toplantı sonucunda verilen kararlar hızlıca yapılmış bir tasarıma uygulanarak yazılımın ilk örneği elde edilir. Bu ilk örnek "prototip" olarak isimlendirilir ve kullanıcılara sunulur. Kullanıcı deneyim ve değerlendirmelerine göre prototip üzerinde değişiklikler yapılır. Prototipin yeni hali de kullanıcılara sunularak yazılımın beklentileri karşılayacak işlevlere sahip ilk sürümü geliştirilmiş olur. Prototip Modelin simgesel gösterimi Şekil 4'te gösterilmektedir.

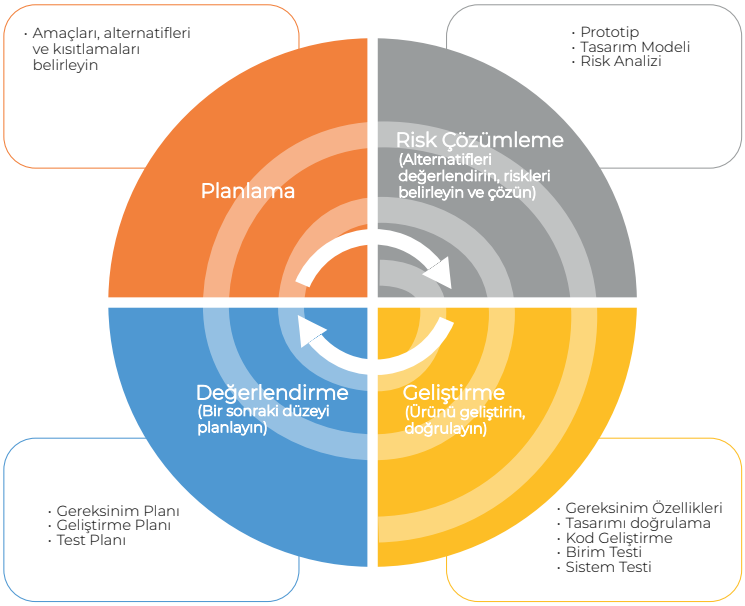


Şekil 4- Prototip Model

Prototip geliştirilmesi esnasında yazılımın gereksinimleri karşıladığı kabul edilene kadar müşteri değerlendirmesine sunulur. Prototip geliştirme tamamlandıktan sonra yazılıma yönelik testler uygulanır ürün tamamlanır. Bu modeli özel yapan nokta müşteri ile geliştirilmesidir.

4. Spiral Model

Spiral model, yazılım geliştirmedeki risklerin azaltılmasına vurgu yapmaktadır. Bu nedenle model, yazılım sürecine risk odaklı bir yaklaşım sergilemektedir. Model proje geliştirme döngülerinden geçerek, proje riskini azaltırken yazılım sistemini aşamalı olarak geliştirmek için döngüsel bir yaklaşım sağlamaktadır. Spiral modelin dört çeyreği vardır ve yazılım projesi aşamalı olarak döngüsel geçiş sergiler. Spiral Modelin simgesel gösterimi Şekil 5'te gösterilmektedir.



Şekil 5- Spiral Model

Şekil 5'te gösterildiği gibi, spiral yol çok düzgün olmayabilir. Her döngü, endişelerin, bileşenlerin veya yapıların her biri için aynı adım dizisini içerir. Yazılım geliştirme ve yazılım geliştirme projelerine eşit derecede uygulanabilir olan spiral model, hedef odaklıdır. Spiral süreç, nihai sonuca ulaşılana veya ulaşılamaz olduğu gösterilene kadar, bu hedefin veya gereksinimin sürekli olarak test edilmesini veya yinelenmesini içerir. Dört kadrandan tipik bir geçiş aşağıdaki gibidir:

1. Spiralin her bir döngüsü için hedefleri, alternatifleri veya kısıtlamaları tanımlanmalıdır.

2. Hedeflere ve kısıtlamalara göre alternatifler değerlendirilmelidir.

3. Tanımlanan risklerin miktarına ve türüne bağlı olarak, bir prototip geliştirilmeli, daha ayrıntılı bir değerlendirme yapılmalı, evrimsel bir gelişme veya belirlenen hedefe ulaşma riskini daha da azaltmak için gerekirse başka bir adım geliştirilmelidir. Öte yandan, risk önemli ölçüde azaltılırsa, bir sonraki adımda gereksinimler, tasarım veya kodlama gibi bir görev belirlenmelidir.

4. Hedefe ulaşıldığı doğrulanmalı ve bir sonraki döngü için plan yapılmalıdır. (Abrahamsson vd., 2002)

B. Çevik Yazılım Geliştirme Modelleri

Bu kategoride yer alan yazılım geliştirme modellerinde yazılım geliştirme süreçlerinin ihtiyaçlarının daha çok dikkate alındığı görülmektedir. Çevik yazılım geliştirme modelleri ayrıca yeni nesil yazıl geliştirme modelleri olarak da adlandırılmaktadır (Sarıdoğan, 2008). Birçok çevik yazılım geliştirme modeli olmakla birlikte bu başlıkta tanıtılacak modeller şunlardır:

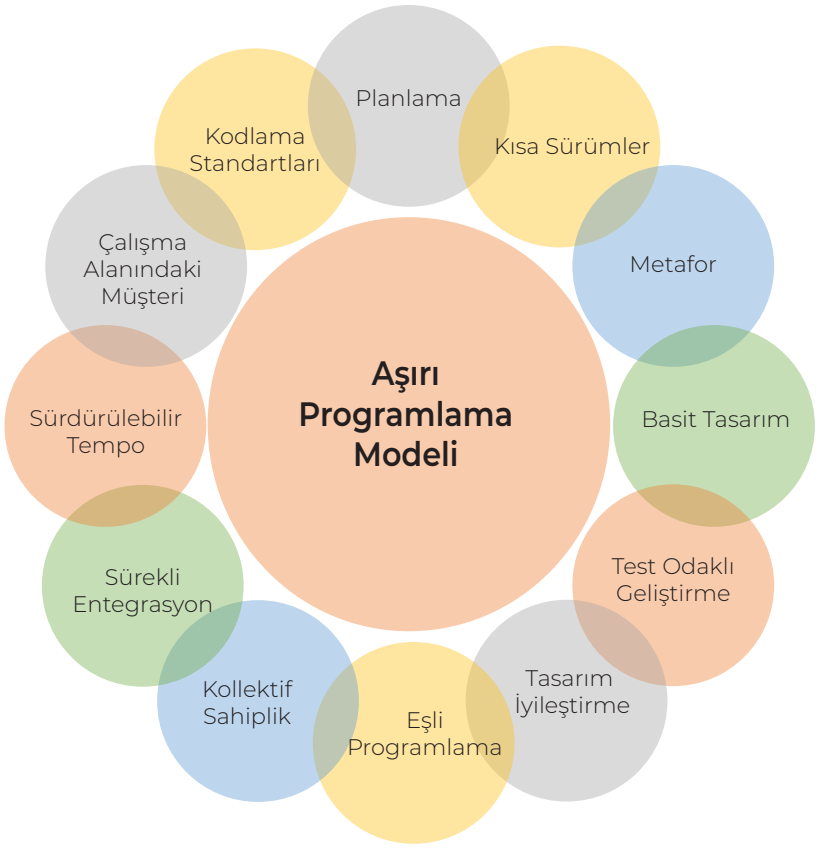
1. Aşırı Programlama Modeli

2. SCRUM Modeli

1. Aşırı Programlama Modeli

Aşırı Programlama (Extreme Programming) geleneksel modellerin uzun geliştirme döngülerinin yarattığı sorunlara bir çözüm getirmek amacıyla yapılan başarılı uygulamalardan doğmuş bir model olarak sunulmaktadır. Aşırı programlama modelinin bireysel uygulamalarının yeni olmadığı belirtilmektedir. Başarılı bir dizi uygulamanın adımları işlevsel olacak şekilde bir araya getirilerek yazılım üreticileri için yeni bir

model olarak sunulmuştur (Abrahamsson vd., 2002, s. 18). Aşırı Programlama Modeli kullanılırken ekip üyeleri ve müşteri arasında sık iletişim kurulmalı, tasarım ve kodda sadelik esas olmalı, birçok farklı düzeyde geri bildirim sağlanmalı ve proje için uygun görünmüyorsa bu modeli kullanmaktan kaçınılmalıdır. Aşırı Programlama Modelinin 12 anahtar kavramı bulunmaktadır (Abrahamsson vd., 2002) ve bunlar Şekil 6'da gösterilmektedir.

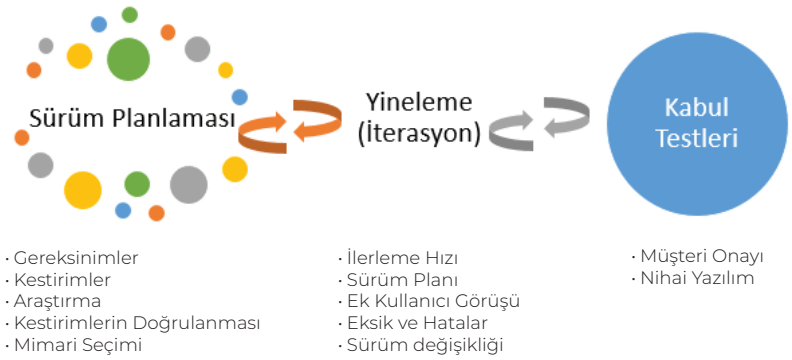


Şekil 6- Aşırı Programlama Modelinin Anahtar Kavramları

Aşırı Programlama Metodu temel olarak üç ana adımdan oluşur (Sarıdoğan, 2008). Kullanıcıdan alınan bilgiler doğrultusunda gereksinimler belirlenir ve ilk aşama olan "*Sürüm Planlaması*" yapılır. Bu aşamada yazılıma yönelik kestirimler ortaya çıkar ve bu öngörülere yönelik araştırma yapılarak bu kestirimler doğrulanmaya çalışılır. Aynı zamanda yazılım projesinin mimarisi de ortaya konur.

İlk aşamadan elde edilen verilere göre ilerleme hızı sürüm planı ve ek kullanıcı bilgileri ile "*Yineleme*" aşamasına geçilir. İkinci aşama kodlamanın yapıldığı aşamadır ve sürekli olarak ilk aşamaya dönülerek yazılımda yapılması gereken değişikliklere yönelik olarak Sürüm Planlaması aşamasında da değişiklikler yapılır.

İkinci aşamanın tamamlanmasının ardından "*Kabul Testleri*" aşamasına geçilir. Üçüncü aşamada yapılan testler sonucu bulunan hatalar veya eksikler ışığında Yineleme aşamasına dönülerek gerekli düzeltmeler yapılır ve düzeltilen yazılım yeniden üçüncü aşamadaki testlere tabi tutulur. Bu aşamada ilk aşamaya girdi olarak alınan kullanıcı gereksinimleri karşılanıp karşılanmadığı da denetlenmektedir. Nihai olarak müşteri onayı alındıktan sonra sürüm kullanılmaya başlanır. Aşırı Programlama Modelinin simgesel gösterimi Şekil 7'de verilmektedir.



Şekil 7- Aşırı Programlama Modeli

2. SCRUM Modeli

Scrum Modeli takım rolleri, olaylar, ürünler ve kurallardan oluşan bileşenlere sahiptir. Bu modelle geliştirilecek bir yazılımda, bileşenlerin tümünü kullanmak gerekir, ancak başka teknikler veya bileşenler de eklenebilmektedir. Scrum Modeli yinelemeli bir yöntemdir ve sprint adı verilen kısa yinelemelere dayanır. Sprint'ler, toplam yazılım ürününün işlevlerine yönelik küçük geliştirme birimleridir. Süreleri kısadır (bir ay veya daha az) ve zaman sınırlıdır (süreleri sabit tutulur, ancak gerekirse kapsam ayarlanabilir). Modelin ideal uygulamalarında, her sprint sonunda potansiyel olarak kullanılabilir bir ürün ortaya çıkar. Scrum Modeli ile geliştirilen bir yazılım projesi paydaşları için üç temel rol tanımlanmakta ve ihtiyaç duyulduğunda bu rollere başka rollerin de eklenebileceği belirtilmektedir (Abrahamsson vd., 2002). Bu roller:

1. Müşteriyi temsil eden ve ekibin işletmeye değer katmasını sağlayan ürün sahibi,

2. Yazılımı üreten, genellikle üç ila on geliştiriciden oluşan geliştirme ekibi,

3. Bir proje lideri olarak değil, bir koç olarak ekibi yönlendiren ve Scrum'un takip edilmesini sağlayan bir Scrum Üstadı.

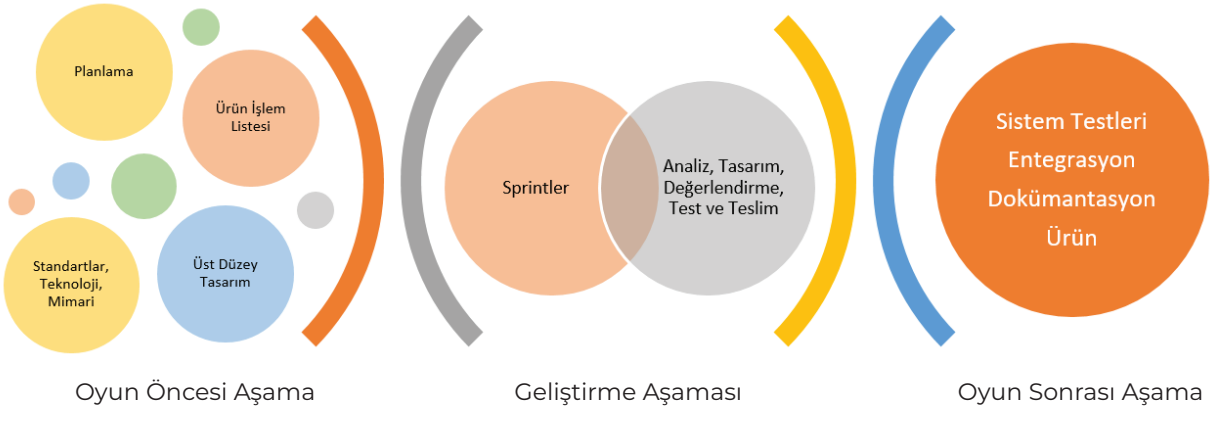
Bunlara ek olarak paydaşlar ve yöneticiler yardımcı rol olarak tanımlanabilir. Scrum, üç temel direk üzerine kurgulanmalıdır, bunlar:

1. Şeffaflık (süreci görünür kılmak),

2. Denetleme (ürün ve ilerlemenin)

3. Adaptasyon (önemli bir sapma tespit edildiğinde, düzeltin).

Scrum Modeli Oyun Öncesi, Geliştirme ve Oyun Sonrası olmak üzere üç ana aşamadan oluşmaktadır (Abrahamsson vd., 2002). Aşamaların simgesel gösterimi Şekil 8'de verilmiştir.



Şekil 8- Scrum Modeli

BÖLÜM ÖZETİ

Günümüzde bilgi sistemleri; eğitim, sağlık, finans, seyahat ve eğlence gibi yaşantımızın neredeyse tüm alanlarında kullandığımız sistemlerdir. Bilgi sistemleri sayesinde birçok iş ve işlem zahmetsiz ve hızlı bir şekilde tamamlanabilmektedir. Bilgi sistemi; doğru zamanda, doğru kişiye, doğru veri ve bilgiyi sağlamayı amaçlayan donanım, yazılım, veri, insan ve prosedürel bileşenler kümesi olarak tanımlamaktadır.

Sistematik olarak geliştirilen projelerin geçirdiği süreçleri açıklayan birçok modelin ortak noktalarını tanımlamak için sıkça "Sistem Geliştirme Yaşam Döngüsü" kavramının kullanıldığı görülmektedir. Bu ortak model temel olarak planlama, analiz, tasarım, uygulama ve değerlendirme adımlarını içermekte ve bir döngü şeklinde uygulanmaktadır.

Planlamada projeye neden ihtiyaç duyulduğu, ne kadar zamanda tamamlanması gerektiği, ne kadar bütçe gerektiği ve projenin kazanımlarının neler olacağı gibi soruların yanıtlanması gerekmektedir. Planlama adımının en kritik yanının projeye devam edip edilmeyeceğine dair kararın çoğunlukla bu aşamanın sonunda verilmesi olduğu düşünülmektedir.

Analiz adımında projenin ihtiyaçları belirlenmeye çalışılmaktadır. Sistemin girdi ve çıktılarının neler olacağı, kimler tarafından kullanılacağı, kullanıcıların beklentilerinin neler olduğu gibi soruların cevaplanması gerekmektedir. Bu tip soruları cevaplamak için de bilgi toplanması, tanımlar yapılması, örnek senaryolar hazırlanması, alternatif çözüm önerilerinin getirilmesi ve tüm bu verilerin değerlendirilmesi gerekmektedir.

Tasarım; analizde elde edilen veriler temelinde geliştirilecek sistemin tasarlandığı adımdır. Bu adımda sistemin nasıl görüneceği ve nasıl çalışacağına dair eskizler hazırlanmaktadır. Kullanıcı ve sistem arayüzünün nasıl olacağı, bileşenlerin birbiriyle nasıl etkileşime gireceği, veritabanı ve sistemle nasıl entegre olacağı, sistem kontrollerinin nasıl yapılacağı gibi başlıklar tasarlanıp görselleştirilmektedir.

Uygulama adımında şekillenen sistemi kurmak hedeflenmekte ve tasarım adımında belirlenen arayüzler, ilişkiler, veritabanı ve kontroller gerçekleştirilmektedir. Bu adımda sistem belirlenen yazılım dilinde kodlanır ve veritabanı yönetim sistemleri işe koşulur.

Değerlendirme; destek, entegrasyon ve test gibi farklı isimlerle de anılmaktadır. Değerlendirme adımının amacı ortaya konulan sistemin doğru çalıştığından emin olmak ve varsa eksik ya da hatalı yönlerinin tespit edilerek düzeltilmesini sağlamaktır. Bu adımda elde edilen veriler sistemin yeniden ele alınması için birinci adımdan itibaren girdi olarak kullanılabilmektedir. Bunun yanında başka adımlara da doğrudan veri sağlayıp, oradaki eksikliklerin giderilmesinde de kullanılabilmektedir.

Kaynakça

Abrahamsson, Pekka., Salo, Outi., Ronkainen, Jussi., & Warsta, J. (2002). Agile Software Development Methods : Review and Analysis. VTT.

Davis, W. S., & Davis, W. S. (Ed.). (1999). The Information System Consultant's Handbook. CRC Press.

Foster, E. C. (2022). Software Engineering; A Methodical Approach (2nd bs). Taylor & Francis.

Sarıdoğan, E. (2008). Yazılım Mühendisliği (2nd bs). Papatya Yayıncılık Eğitim.

Şeker, S. E. (2015). Yazılım Geliştirme Modelleri ve Sistem/Yazılım Yaşam Döngüsü. YBS Asiklopedisi, 2(3). https://ybsansiklopedi.com/

Silahtaroğlu, G. (2010). Bilgisayar ve Yazılım Mühendisliğinde Sistem Analizi ve Tasarımı. Papatya Yayıncılık Eğitim.

Tsui, F., Karam, O., & Bernal, B. (2017). Essentials of Software Engineering (4th bs). Jones & Bartlett Learning.

Wasson, C. S. (2006). System Analysis, Design, and Development : Concepts, Principles, and Practices. John Wiley & Sons.