BLM 426 YAZILIM MÜHENDİSLİĞİ

BAHAR 2016

Yrd. Doç. Dr. Nesrin AYDIN ATASOY

3. HAFTA: YAZILIM İSTERLERİ ÇÖZÜMLEMESİ

Yazılım geliştirme sürecinin başarısı için gereksinimlerin ve yazılım isterlerinin çok iyi anlaşılması gerekir. Bir bilgisayar programı ne kadar iyi tasarlanmış ve yazılmış olursa olsun müşteri isteklerini tam olarak karşılamıyorsa başarılı olmuş sayılamaz. Bu nedenle, sistem isterleri belirlendikten sonraki yazılım geliştirme sürecinin başlangıcında yer alan yazılım isterleri çözümlemesi aşamasında, müşterinin yazılımdan beklentileri belirlenir, gereksinimler açıklığa kavuşturulur, yazılım isterleri modellenir ve tanımlanarak sonraki aşamalar için bir temel oluşturulur.

İsterler çözümlemesi aşamasında hem müşteri hem de geliştirici aktif rollere sahiptirler. Eğer bu iki taraf geniş ölçekli birer grup iseler ve bu aşamada görev alan kişiler yeterli deneyime sahip değillerse, yapılan çözümleme de eksiklik olabilecek ve sonraki aşamalarda sorunlar yaşanması muhtemel olacaktır. Böyle bir olumsuz duruma yer vermemek için hem müşteri hem de geliştirici tarafından bu aşama hafife alınmadan gereken özen gösterilmelidir.

➢ İster Nedir?

İster (gereksinim): Gerekli olan, istenen veya ihtiyaç duyulan anlamına gelmektedir.

İster Tanımı: Bir sistemin gereksinimleri, o sistem tarafından sağlanan hizmetlerin ve işlevsel kısıtların tanımıdır.

IEEE 729: Kullanıcı tarafından bir problemi çözme ya da bir hedefi gerçekleştirmek için ihtiyaç duyulan durum ya da destek.

Gereksinim, sistemin ya da işlevlerinin nasıl yerine getirileceği ile ilgili değildir. Ne olduğu ile ilgilidir.

- hangi veri tabanı,
- hangi tablolar,
- ne kadar bellek kullanılıyor,

bunlar gerçekleştirim aşamasında ele alınır.

- **İster Mühendisliği Nedir?**

Tüm bu hizmet ve kısıtlamaların,

- Belirlenmesi,
- Çözümlenmesi,
- Belgelendirilmesi
- Kontrol edilmesi

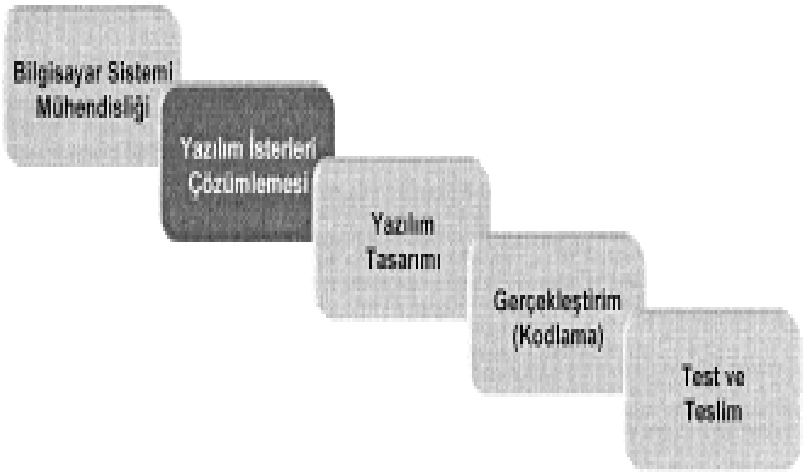
Sürecine İster (Gereksinim) Mühendisliği denir.

- **İsterler neden önemlidir?**

İsterlerden kaynaklı hatalar geç aşamalarda fark edilir ve genellikle yanlış bilgi, ihmal ve tutarsızlıktan kaynaklanır. Bu durumda da düzeltilme maliyetleri yüksek olur.

- **İster Çözümleme Aşaması**

Yazılım mühendisliği etkinliği olarak isterlerin çözümlenmesi Şekil 1' de görüldüğü gibi bir geçiş sağlar. Bu çözümleme, sistem mühendisine yazılımdan beklediklerini tanımlama olanağı sağlarken; yazılım mühendisine de yazılım öğelerinin ortaya çıkması, süreçler arasındaki veri ve denetim akışlarının tanımlanması için olanak sağlar.



Şekil 1. İsterler çözümlemesinin yeri.

İsterler çözümlemesi, yazılım mühendisliği sürecinin ilk teknik aşamasıdır. Bu aşamada, **yazılımın kapsamı belirlenir, bundan sonraki geliştirme aşamalarında kullanılacak tanımlamalar oluşturulur.** Çözümleme sırasında, problem alanındaki bilgi akışına, işlevselliğe ve davranış konularına ağırlık verilir. İstenirse bir ilk örnek oluşturulabilir ve çözümleme sonunda yazılım isterlerinin olduğu bir belge hazırlanır. Bu belge, geliştirici ve müşteri tarafından incelenerek onaylanır.

- **Çözümleme Çalışmaları**

Yazılım isterleri çözümlemesi, çözümleyici adı verilen yeterince deneyimli kişiler tarafından gerçekleştirilmelidir. Çözümleme çalışmaları beş başlık altında toplanarak incelenebilir:

1) **Problemin Anlaşılması:** Çözümleyici, Sistem Belirtimi Belgesi'ni inceleyerek problemin ne olduğunu, bunun için gerekli yazılımın kapsamını belirlemeye çalışır.

2) **Problemin Çözümlenmesi:** Çözümleyici, mevcut sistemi inceleyerek topladığı bilgilere göre, sistemi etkileyen giriş/çıkış etkinliklerini ve kısıtlamaları dikkate alarak yazılım işlevlerini tanımlar. Yazılımın ne yapması gerektiğini ortaya koyar.

3) **Modelleme:** Değerlendirmeler ve çözümlemeler sırasında çözümleyici, sistemin çalışma şeklini, veri akışını, işlevselliği daha iyi anlayabilmek için modeller oluşturur. Bu modeller kağıt üzerinde çizili diyagramlar, şekiller olabileceği gibi çalışan programlar da olabilirler.

4) **Belirtim:** Tüm çözümlemeler ve modeller, müşteri ve geliştiricinin üzerinde anlaştığı bir yazılım belirtimi ortaya çıkarmayı amaçlar. Aslında sistem ve yazılım isterlerinin müşteri tarafından hazırlanması gerekir. Ancak çoğu zaman bu iş müşteri ve geliştirici ile beraber yapılır. Bu aşamada kullanıcının sistemi nasıl kullanacağını ortaya koyan taslak bir kullanım kılavuzu da hazırlanır ve özellikle insan-makine arayüzü içeren sistemlerde kullanıcının beklentileri dikkate alınarak sistemin tasarımı yapılır. Temel işlevler, kısıtlar, çalışma şekilleri, ara yüzler, başarım ölçütleri, doğrulama ve geçerleme yöntemleri tanımlanır.

5) **Gözden geçirme:** Çözümleme aşaması sonunda ortaya çıkan belgeler müşteri ile birlikte gözden geçirilir. Bu gözden geçirme sonunda değişiklikler yapılsa bile çözümleyici ve müşterinin anlaştığı **Sistem İsterleri Belirtim** belgesi ortaya çıkar.

- **İsterlerin Değişmesi**

İsterlerin çözümlenmesi çok iyi yapılmasına rağmen süreç esnasında isterlerde değişiklikler meydana gelebilir. Bunun sebepleri:

- Müşteri ve geliştirici arasındaki iletişimin yeterli olmaması,
- İsterlerin çözümlenmesi aşamasını çabuk geçebilmek için bazı varsayım ya da kabullenmeler yapılmış olması,
- Müşterinin ne istediğini tam bilememesi ve sık sık fikir değiştirmesi,
- Geliştiricinin deneyim eksikliği,
- Ayrıntılı tasarıma geçilince yeni isterlerin gerekliliğinin ortaya çıkması

olabilir. Unutulmamalıdır ki ne kadar değişiklik olursa olsun tüm isterlerden müşterinin haberi olmalı ve onaylı belge haline getirilmelidir.

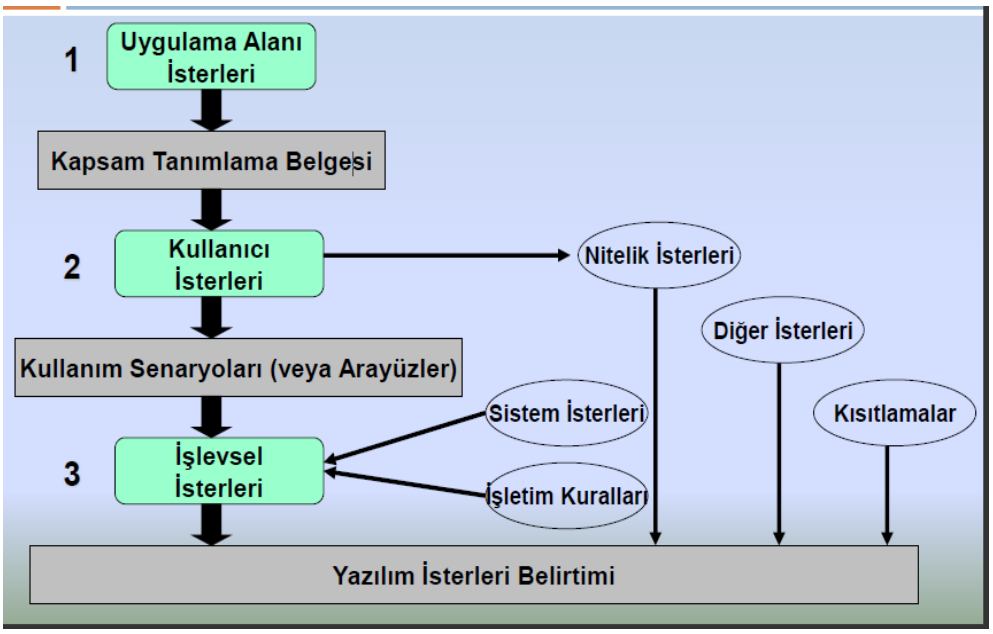
- **İsterlerin Belirlenmesi**

Sistemin başarısı, sistemden ne istendiğinin doğru olarak algılanmasına bağlıdır. Bunun için düzeylere ayrılmış sistem isterlerinden Yazılım İsterleri Belirtimi (System Requirement Specifications) çıkartılmalıdır.

➢ **İsterlerin Düzeyleri**

İsterlerin belirli düzeylere bölünerek toplanması özellikle büyük sistemlerde çok yarar sağlar.

- **Düzey 1:** Uygulama alanı isterleri
- **Düzey 2:** Kullanıcı isterleri
- **Düzey 3:** İşlevsel isterler



Şekil 2. İsterler düzeyleri.

Düzey 1: Uygulama alanı isterleri: Müşteri ile çeşitli kanallar yardımıyla iletişimde bulunulur. Kullanım ortamında gözlem yaparak nelere gereksinim duyulduğu ve işin kapsamı belirlenir.

Düzey 2: Kullanıcı isterleri: Bu düzeyde, işletmen, yani sistemi kullanan kişiler için gereksinimler dikkate alınır; kullanım senaryoları, kullanıcı ara yüzleri ve temel nitelik etmenleri tanımlanır.

Düzey 3: İşlevsel isterler: Bu düzeyde sistemin bir bütün olarak genel isterleri ele alınır; işletim kuralları belirlenir. Müşteri ile birlikte yapılan toplantılarda ne istendiği, işlevlerin neler olacağı, sistemin nasıl kullanılacağı, elle yapılması gereken işlerin ne olacağı tam olarak belirlenir.

➢ **İster Belirleme Yöntemleri**

İsterler belirlenirken izlenebilecek yöntemler arasında şunlar vardır:

- Müşterinin verdiği öneme göre önceliklendirme
- Sınıflandırma
- Kaydetme
- Numaralandırma
- İsterin sürümlerini saklama

- **İster Tanımlanması**

İsterler çözümlemesi sırasında üretilen belirtim belgesinde isterleri tanımlamak üzere çeşitli teknikler kullanılır. Bunlardan en yaygın olanlar şunlardır:

- Kullanım Senaryoları
- Veri Akış Diyagramları(VAD)
- Varlık İlişki Diyagramları
- Nesne Diyagramları
- İşlevsellik Diyagramları
- Etkileşim Diyagramları
- Durum Haritası Diyagramları
- Durum Geçiş Diyagramları
- Resmi Belirtim Dilleri
- Veri Sözlüğü

- **İsterlerin Sınıflandırılması**

İsterlerin sınıflandırılması, isterlerin daha iyi anlaşılmalarını ve iyi tanımlanmalarını sağlar. Belirtim hazırlanırken dikkate alınması gereken sistem özelliklerini şöyle listeleyebiliriz:

İşlevsel Özellikler

- Veri işleme özellikleri
- İşlem hızı, tepki süresi, işlem kapasitesi
- Hesaplama doğruluğu

Güvenlik

- Kesintisiz kullanım ve güvenilirlik
- Erişim güvenliği

Kullanım kolaylığı

- Kullanıcı dostu ara yüzler
- Öğrenme kolaylığı
- Uygun kullanım kılavuzları
- Diğer yazılımlar ile ara yüzler
- İşletim sistemi veya diğer sistemler ile ara yüzler

Bakım kolaylığı

- Platform arası taşınabilirlik
- Başka sistemlerle beraber çalışabilirlik

Teknik belgelendirme

Veri yedekleme ve kurtarma özellikleri

Geliştirme dili, aracı, tekniği veya yöntembilimi ile ilgili kısıtlar

- **Çalışma Alanının Anlaşılması**

Çalışılacak alan çok iyi anlaşılmalı ve bilgiler özümsenmelidir. Her türlü yazılımın aslında veri işlemeye dayandığı düşünülürse, verilerin en önemli unsurlarından biri olduğu anlaşılacaktır. Yazılım veriyi bir şekilde girdi olarak alır, işleyerek bir başka biçime dönüştürür; sonrada ya çıktı olarak dışarı verir ya da bir alt sisteme kumanda edecek işaretlere dönüşmesini sağlar. Verilerin yanında, yazılım, açma, kapama gibi iki durumlu olaylara da yanıt vermek durumundadır. Böylece çalışma alanındaki bilgiler veri ve olay olarak tanımlanabilmektedir.

Çalışma alanının anlaşılması ile ilgili yöntemler:

1) **Sorma**

Kullanıcı isteklerini öğrenmenin en iyi yolu sormaktır. Sorular karşılıklı görüşme ya da anket yoluyla sorulur. Karşılıklı görüşme sırasında isterlerle ilgili amaçlar, düşünceler, talepler araştırılır. Çok sayıda kullanıcının görüşlerini toplamak amacıyla anketler yapılabilir. Anket sonuçlarına göre genel sorunlar, istekler belirlenir ortalama değerler elde edilir.

2) **Modelleme**

Yazılım geliştirmeden önce, sistemin yapması gerekenleri gösteren, insan-makine ara yüzünü grafik olarak gerçekleştiren modeller kullanılmalıdır. Modeller çalışan programlar şeklinde olabilir ya da kağıt üzerinde metin destekli çizimlerden oluşturulabilir.

3) **Ayrıştırma**

Büyük ve karmaşık sistemlerin bir bütün olarak anlaşılması ve tanımlanması güç olacağından, sistem daha küçük ve anlaşılabilir kısımlara bölünmeli ve aralarındaki ara yüz tanımlanarak bütün sistemin işlevselliği kapsanmalıdır.

- **Çözümleme Yöntemleri**

Son yıllarda çeşitli çözümleme yöntemleri ortaya çıkmış olmasına rağmen, her çözümleyici için temel sayılabilecek bazı temel ilkeler vardır. Bu ilkeler şu şekilde sıralanabilir:

- ✓ **Yapısal Çözümleme**

Hem çözümlemede hem tasarımda kullanılabilen bir yöntemdir. Veri akışı ve denetimi gösterilerek sistemin işlevlerinin ve davranışının modellenmesine dayanır.

1) Veri Akış Diyagramı
2) Davranış Modellemesi
    - Süreç Etkinleştirme Tablosu
    - Durum Geçiş Diyagramı

    - Süreç Belirtimi
    - Karar Tabloları
    - Varlık İlişki Diyagramı
- Veri Sözlüğü

Veri Akış Diyagramı (VAD) : Sistem içindeki her verinin nasıl taşındığı ve bu veri akışını sağlayan fonksiyonların neler olduğu veri akış diyagramında tarif edilir. Veri akış diyagramı; sistemin varlıkları, süreçleri, istemdeki veri depoları ve bunlar arasındaki verinin nasıl aktığını gösterir.

Bilgi bilgisayar sistemi içinde akarken çeşitli dönüşümlere uğrar. Sistem çeşitli formlarda girdi alır ve bu girdileri yazılım, donanım ve insan elemanları ile işleyerek çeşitli formdaki çıktılara dönüştürür. VAD verinin girişten çıkışa kadar olan dönüşümü ve bilginin taşınmasını gösteren grafiksel bir tekniktir.

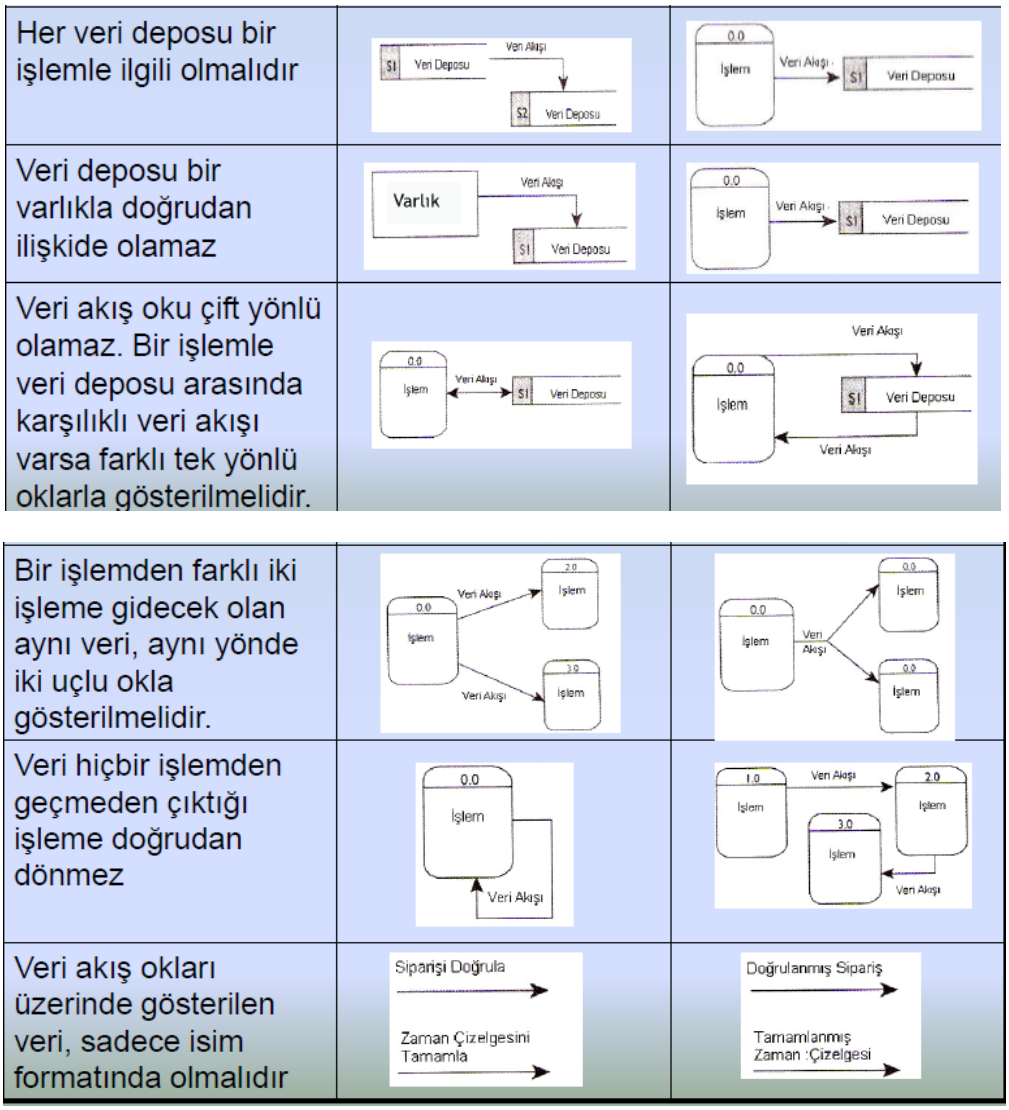
VAD Simgeleri

Anlam	Simge - 1	Simge - 2	Örnek
Dış varlık			Öğrenci
İşlem (süreç)			1.1 Yeni Öğrenci Kaydı
Veri akışı			Yeni Öğrenci Bilgisi
Veri deposu	Veri deposu		D1 Öğrenciler

Şekil 3. VAD simgeleri.

VAD Kuralları

Kural	Yanlış	Doğru
İşlemin sadece çıkışı olamaz.	0.0 İşlem — Veri Akışı	Veri Akışı — 0.0 İşlem — Veri Akışı
İşlemin sadece girişi olamaz.	Veri Akışı — 0.0 İşlem	Veri Akışı — 0.0 İşlem — Veri Akışı
İşlem girişleri istenen çıkışı verecek kadar yeterli olmalıdır.	S1 Çalışma Dosyası, Çalışma Saatleri — 1.0 Maaş Hesaplama	S1 Çalışma Dosyası, Çalışma Saatleri, Çalışan Verisi — 1.0 Maaş Hesaplama — S2 Çalışan Dosyası



Şekil 4. VAD kuralları.

VAD Düzeyleri

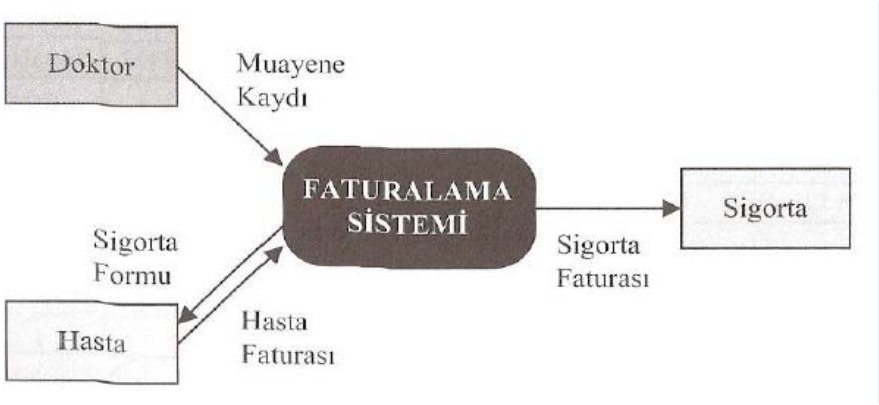
VAD bir sitemi ya da yazılı herhangi bir soyutlama düzeyinde göstermek için kullanılabilir. VAD artan bilgi akışı ve detayları içerecek şekilde çeşitli seviyeler bölünebilir.

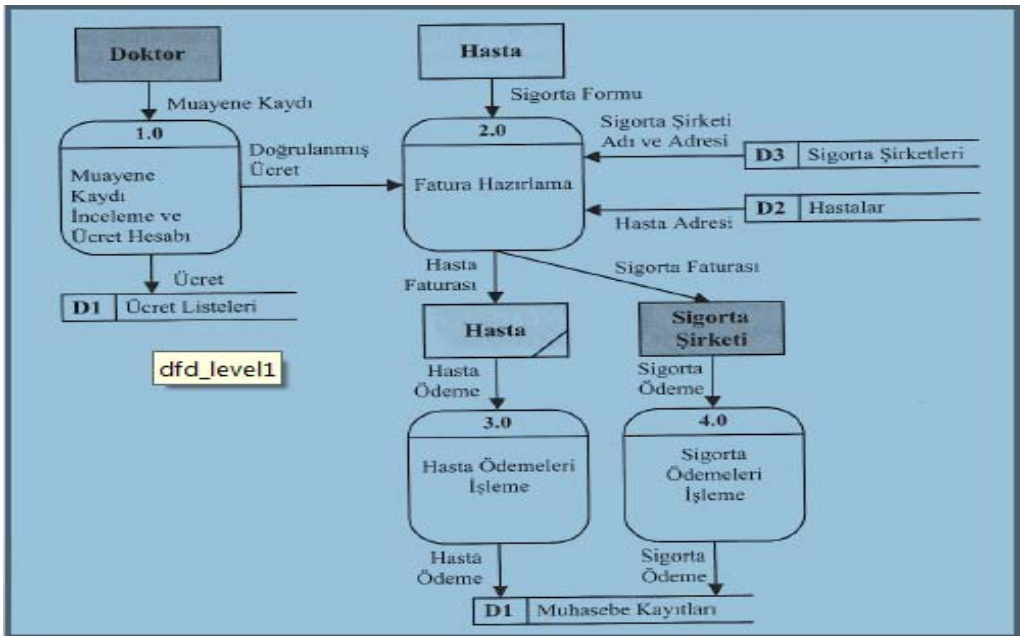
- Düzey 0 olarak gösterilen VAD aynı zamanda kapsam diyagramı (temel sistem modeli) olarak da adlandırılır. Tüm sistem tek bir süreç içerisinde gösterilerek girdi ve çıktılar gelen ve çıkan oklar ile ifade edilirler.

Düzey 0 olan VAD daha detaylı bilgi akışı ve süreçleri içerecek şekilde ek süreçlere ayrılır.

- Düzey 1 VAD 5 ya da 6 süreç ve bunlar arasındaki akışı gösterir.
- Düzey 1 de gösterilen süreçler kapsam modelinde yer alan ana sistemin alt fonksiyonlarını içerir.

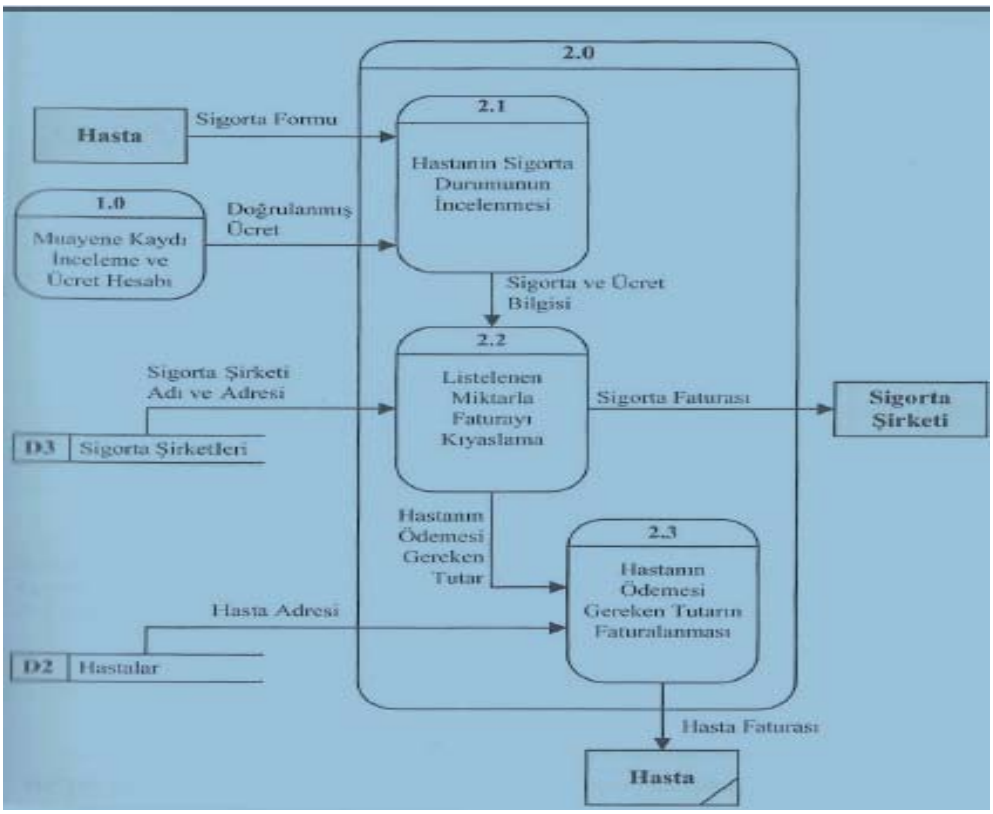
VAD Örneği:





Şekil 5. Düzey 1 Diyagram.

.



Şekil 6. Düzey 2 Diyagram.

- **Davranış Modellemesi**

Her türlü çözümleme yönteminde davranış modellemesi yapmak uygulanan temel ilkelerden biridir. Davranış modelinde sistemin durumları ve bu durumlar arasında geçişlere neden olan olaylar gösterilir.

Bazı çözümleme yöntemleri veri akış diyagramı ile denetim diyagramını ayrı ayrı oluştururlar. Veri akışındaki süreç modelinde bir veri girdisi bir denetim çıkışına neden oluyorsa bir veri koşulu oluşmuş olur. Bu durum, akış modelinde bir denetim belirtimi halinde gösterilir. Bir davranış modeli olarak kullanılan denetim belirtimi, bir süreç etkinleştirme tablosu, bir durum geçiş diyagramı ve sözde kod şeklinde bir açıklama içerir.

Süreç Etkinleştirme Tablosu

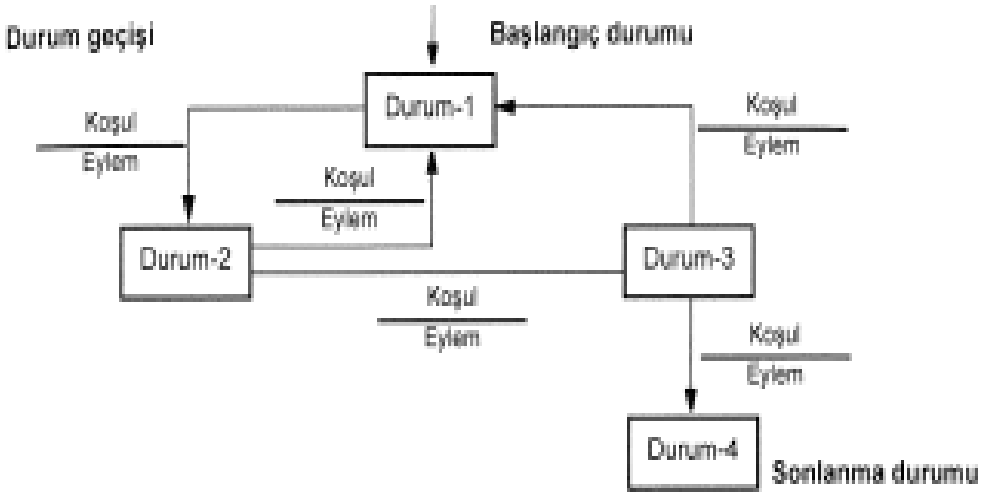
Karar tablosun bir türü olan süreç etkinleştirme tablosu, durum geçiş diyagramı ile beraberce oluşturulur. Durum geçişinde olay oluştuğu zaman akış modelindeki hangi sürecin etkinleşeceğini gösteren bir matris şeklindedir. Düşey olarak durum geçiş koşulu oluştuğunda yapılacak eylemler, yatay olarak da etkinleştirilecek süreçler bulunur. Genellikle “0” etkilenmeyecek süreci, “1”etkinleştirilecek süreci göstermek için kullanılır.

Eylem	Etkinleşen Süreç		
	Süreç-1	Süreç-2	Süreç-3
Eylem-A	0	1	0
Eylem-B	1	0	0
Eylem-C	0	0	1
Eylem-D	0	0	0

Durum Geçiş Diyagramı

Durum geçiş diyagramı, çeşitli durumları ya da çalışma kipleri bulunan her türlü sistemin çözümlemesi için kullanılabileceği gibi özellikle ani tepkileri olan gerçek zamanlı sistemlerin çözümlenmesinde neyin ne zaman olacağını anlatarak hem geliştiriciye hem de belgeleri okuyan kullanıcıya yarar sağlar.

Her diyagram bir başlangıç ile başlar ve mutlaka bir sonlanma durumu vardır. Durumlar ya da kipler bir kutu şeklinde, geçişler de birer ok ile gösterilir. Her duruma mutlaka erişmeli, her durumdan çıkış mutlaka bulunmalıdır. Durum geçişlerine neden olan ve sistem tarafından fark edilen koşul ve bu geçiş sonrasında sistemin gerçekleştireceği eylem, geçişi gösteren okun yanındaki yatay bir çizginin alt ve üstünde belirtilir. Bir durumdan başka bir duruma olabilecek tüm geçişler aynı diyagramda gösterilir. Karmaşık sistemlerde çok sayıdaki durum geçişlerini gruplayarak düzeyler halinde göstermek de mümkündür.



Süreç Belirtimi

Süreç belirtimi; veri ve denetim akış diyagramlarının en son düzeyindeki süreçlerin iç yapılarının tanımlanmasında kullanılır. Genellikle, bir program tanımlama dili kullanılarak yapılan sözde kod şeklinde bir tanımlamadır. Bazen de, tamamen düz, metinsel bir anlatım kullanılarak süreç tamamlanır.

Her süreç için kullanılacak algoritmalar, formüller, tablo ve diyagramlar bu belirtimde yer alabilir.

Süreç belirtimlerinde kullanılacak dilin Türkçe veya İngilizce olması önemli bir karardır. Eğer tasarım Türkçe yapılıp kodlama İngilizce yapılacaksa, bu belirtimin Türkçe olması ve bir çeviri tablosunun yer alması gereklidir.

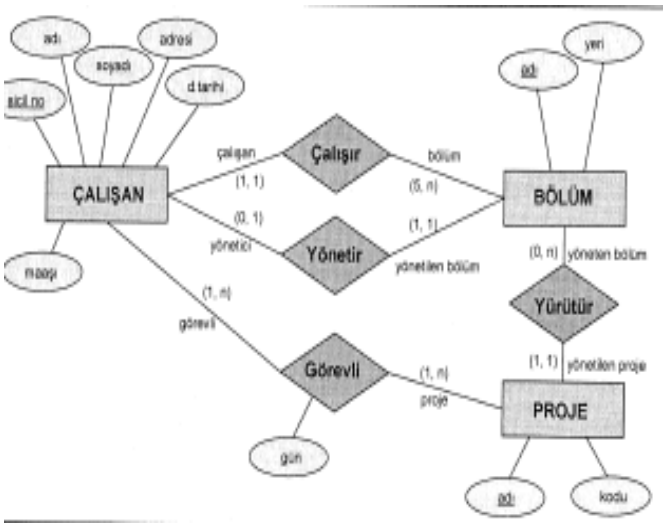
Karar Tabloları

Bazen metinsel açıklamalar süreç belirtim için yeterli olmayabilir. Özellikle karmaşık kararlara göre bir takım çıktılar üretilecekse ve bu kararlar çeşitli değişkenlerin durumlarına göre alınacakasa, bunu düz metin olarak ifade etmeye çalışmak anlaşılabilirliği azaltır. Bu amaçla, yazılımın akışına etki edecek dallanmaları ve koşulları daha düzenli bir şekilde göstermek için karar tablosu kullanılır. Tablodaki satırlara koşullar, sütunlara da yapılacak işlem ve kurallara ait numaralar yerleştirilir. Her koşula ilişkin kural Evet-Hayır, Doğru-Yanlış ya da koşulun yanıtı şeklinde işaretlenir. Bazen de çeşitli sayısal değerler girilebilir.

Kurallar / Koşullar	1	2	3	4	5
Koşul-1	E	H	H	H	H
Koşul-2	H	H	E	H	E
Koşul-3	100	100	200	400	500
...					
Koşul-N			x		

Varlık İlişki Diyagramı

Büyük sistemlerde kullanılan verilerin belirli bir düzen içinde tanımlanması, hangi varlığın hangi varlıkla ilişkisi olduğunun gösterilmesi gereklidir. Bu ilişkileri göstermek, veri deposunda varlıkları ayırt etmek için kullanılacak anahtarları tanımlamak için veri depolarında bulunan veri yapıları aşağıda örneği verilen varlık ilişki diyagramı ile gösterilir.



Varlıklar ve aralarındaki ilişkiler en az ve en fazla olarak bilinen aşağıdaki kurallarla tanımlanır:

- 1:Bir adet A varlığı yalnızca bir adet B varlığıyla ilişkilidir.
- 1-n:Bir adet A varlığı birden fazla B varlığıyla ilişkilidir.
- m-n:Birden fazla A varlığı birden fazla B varlığıyla ilişkilidir.
- 1-3:Bir adet A varlığı 3 adet B varlığıyla ilişkilidir.

- **Veri Sözlüğü**

Yapısal çözümlemenin bir parçası olarak, sistemde kullanılan tüm nesneler, diğer bir deyişle akışı olan veriler, bir listede toplanarak veri sözlüğü oluşturulur. Sözlük genellikle bir otomatik araç tarafından oluşturulur. Her bir giriş için aşağıdaki gibi bilgiler bulunur:

- Bir özel ve tek isim
- İçerik anlatımı
- Nasıl ve nerede kullanılacağı
- Verinin, sayısal ve özel gibi tanımlı tipi
- Varsayılan değer
- Alt ve üst sınır değerleri

Bazı araçlar yardımıyla, veri akış diyagramları ile modellenmiş yazılım öğeleri, yazılım bileşenleri ve yazılım birimleri arasındaki arayüz bilgileri otomatik olarak türetilebilir. Bu bilgiler içinde yer alan veri tanımlamaları veri sözlüğünü oluşturur.

KAYNAKLAR

1. Yazılım Mühendisliği Ders Notları; Yrd.Doç.Dr. Buket Doğan.
2. Yazılım Mühendisliği; M. Erhan Sarıdoğan