



Gereksinim Mühendisliği

BBS-651 Yazılım Mühendisliği

Hafta #3 (20 Ekim 2010)

■ Gereksinimler

- ▶ Gereksinim nedir, türleri nelerdir?
- ▶ Gereksinim tanımlama

■ Gereksinim Mühendisliği Süreci

- ▶ Gereksinim çıkarma, analiz, geçerleme, yönetim

■ İş Gereksinimleri Analizi ve Gereksinim Analizi İlişkisi

- ▶ İş süreçlerini modelleme, gereksinim analizi

■ Sistem Modelleri

- ▶ Bağlam (“context”), süreç (“process”), davranış (“behavior”), veri (“data”), nesne (“object”) modelleri



Gereksinim Türleri ve Gereksinim Tanımlama

Gereksinim Mühendisliği

- **Gereksinim mühendisliği**; müşterinin sistemden istediği servisleri ve sistemin altında çalışacağı kısıtları ortaya çıkarma ve tanımlama sürecidir.
- **Gereksinimler**; gereksinim mühendisliği süreci boyunca ortaya çıkan sistem servislerinin ve kısıtlarının bir tanımıdır.

Gereksinim Nedir?

- Bir sistem servisinin / kısıtının üst seviyeli ifadesinden detaylı matematiksel tanımına kadar farklılık gösterebilir.
 - ▶ Sözleşme için teklife çağrı belgesine esas olabilir – farklı çözümlerin önerilmesine açık olmalıdır.
 - ▶ Sözleşmeye esas olabilir – detaylı olarak tanımlanmalıdır.
- Geliştirilecek sistemin sağlaması gereken bir durum veya yetenek
- Sistemin bir iş ihtiyacını karşılaması için göstermesi gereken özellik
- ...

Kullanıcı ve Sistem Gereksinimleri

■ Kullanıcı gereksinimleri

- ▶ Doğal dilde ifade edilir.
- ▶ Sistemin sağlayacağı servislerin ve uyacağı kısıtların ifadesidir.
- ▶ Müşteriler için yazılır.

■ Sistem gereksinimleri

- ▶ Sistem işlevlerinin, servislerinin ve kısıtlarının detaylı tanımlarını içeren yapısal bir belge ile ifade edilir.
- ▶ Müşteri ve sağlayıcı arasındaki sözleşmenin bir bölümü olarak, sistemin neleri gerçekleştireceğini tanımlar.

Kullanıcı ve Sistem Gereksinimleri: Örnek

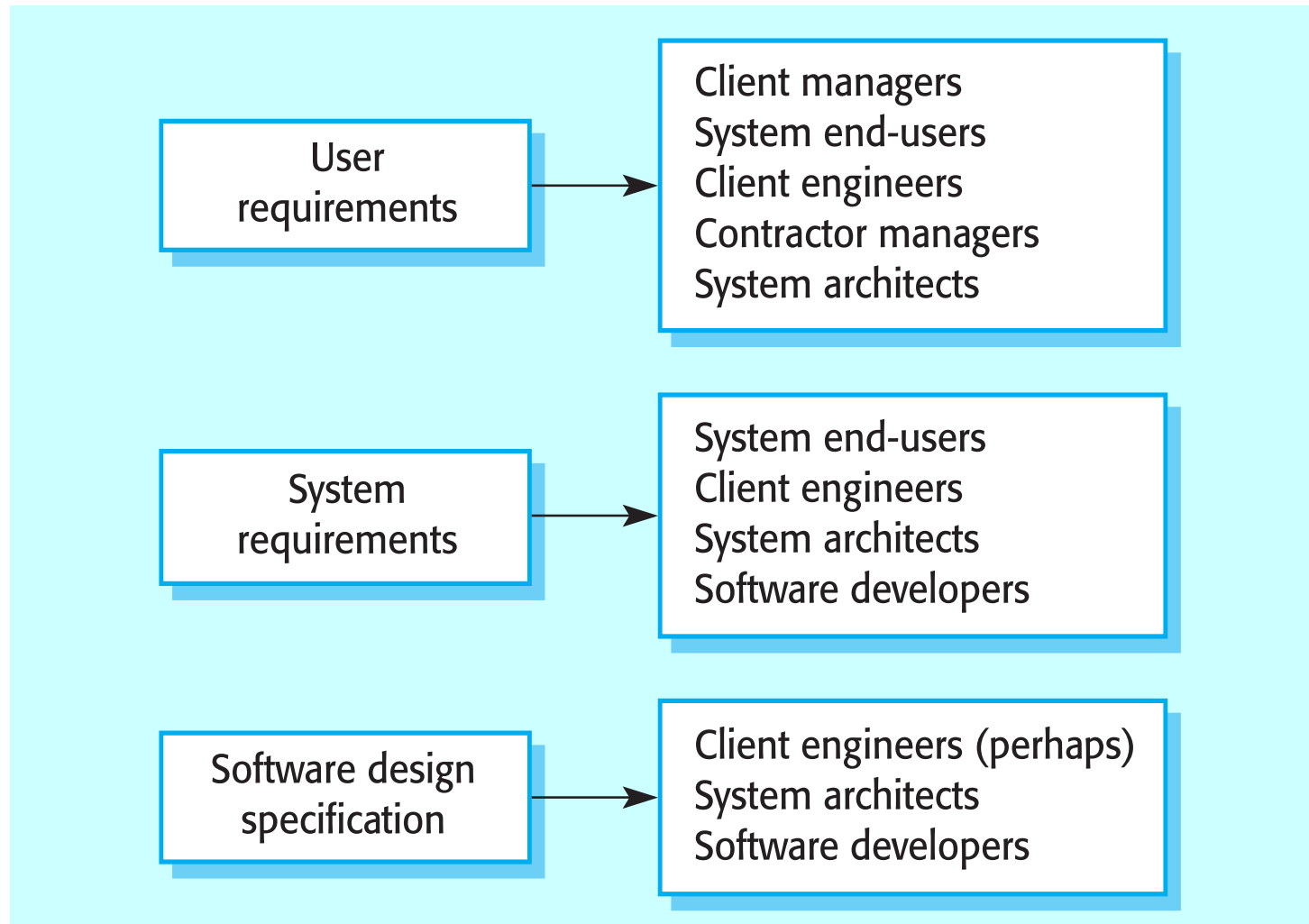
User requirement definition

1. The software must provide a means of representing and accessing external files created by other tools.

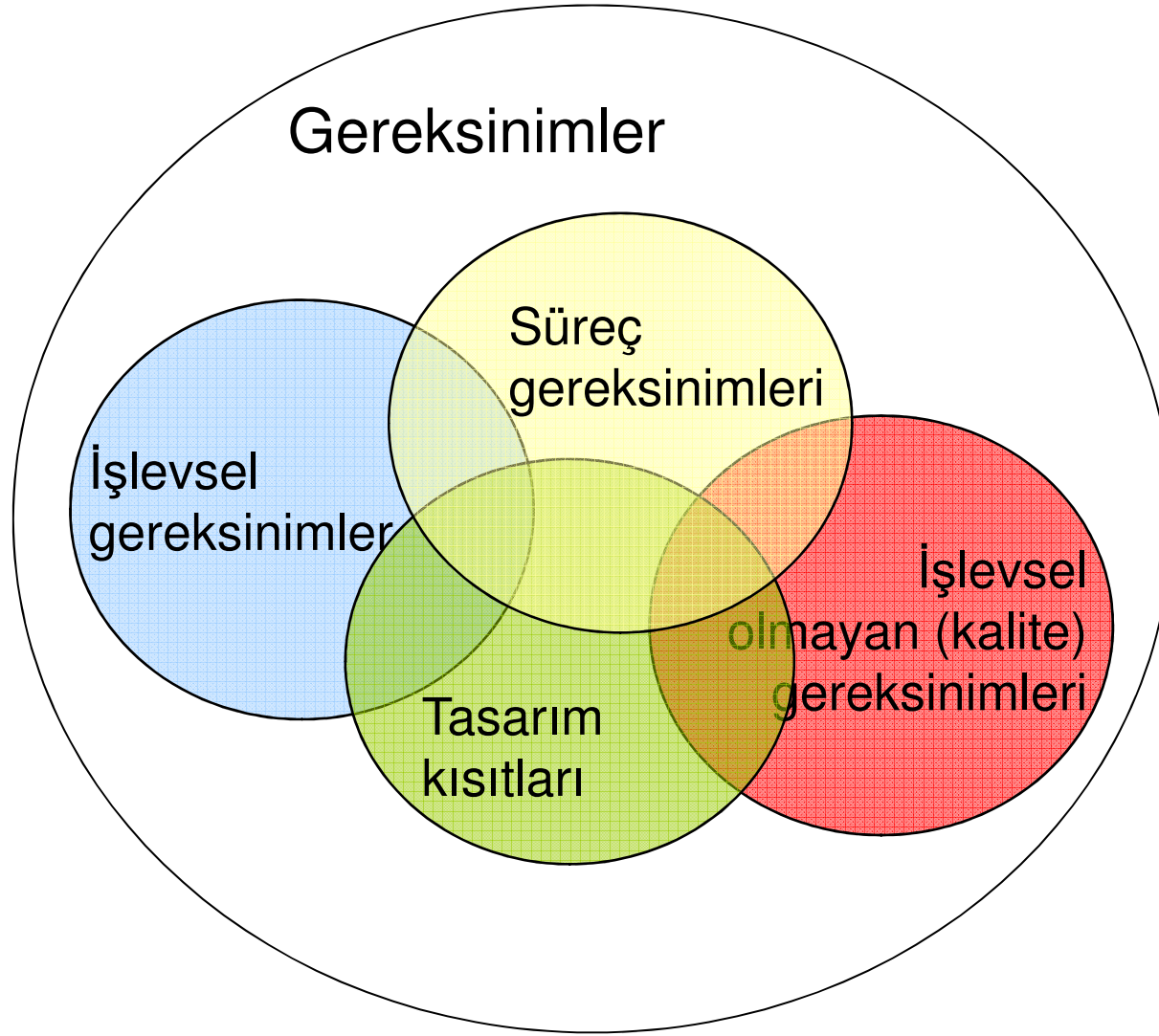
System requirements specification

- 1.1 The user should be provided with facilities to define the type of external files.
- 1.2 Each external file type may have an associated tool which may be applied to the file.
- 1.3 Each external file type may be represented as a specific icon on the user's display.
- 1.4 Facilities should be provided for the icon representing an external file type to be defined by the user.
- 1.5 When a user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

Gereksinimler ve İlişkili Kişiler



Gereksinim Uzayı



Gereksinim Türleri (1)

■ İşlevsel gereksinimler

- ▶ Kullanıcıların sistem ile gerçekleştirmeyi istediği işlemler ve bu işlemlerin özellikleri
- ▶ Sistem “ne” yapacak ?
 - ◆ Girdilerin ve çıktıların tanımı
 - ◆ Girdileri çıktılara dönüştüren işlemlerin tanımı

Gereksinim Türleri (2)

■ İşlevsel olmayan gereksinimler

► Ürün kalite kriterleri

- ◆ Performans, güvenilirlik, kullanılabilirlik vb.
- ◆ Örnek: Hatalar arası ortalama zaman
 - “Bir sistem ya da sistem ögesi için hata oluşumları arasındaki ortalama zamandır.”
 - “Tanımlı bir işletim süresince oluşan hatalar sayılarak ve işletim süresi bu hata sayısına bölünerek hesaplanır.”
 - » $H.A.O.Z. = \text{işletim süresi} / \text{işletim süresince oluşan hata sayısı}$
 - “Sistemin kesin kabul testi süresince ölçülen tüm sistem unsurları için hatalar arası ortalama zamanı en az 60 (altmış) saat olacaktır.”

Gereksinim Türleri (3)

■ Süreç gereksinimleri

- ▶ Geliştirme adımları ile ilgili istekler
- ▶ Tanımlı yaşam döngüsü, kalite güvence etkinlikleri, vb.

■ Tasarım kısıtları

- ▶ Tasarımı etkileyecek istekler
- ▶ Geliştirme ortamı (örnek: J2EE), ilişkisel veri tabanı, vb.

Gereksinimlerin Bulanıklığı

- Gereksinimler net olarak ifade edilmediği zaman problemler yaşanır.
- Bulanık gereksinimler geliştiriciler ve kullanıcılar tarafından farklı algılanabilir.
 - ▶ Örnek: “Sistem, kullanıcının doküman ambarındaki dokümanları okuması için, **uygun görüntüleyiciler** sağlamalıdır.”
 - ▶ “uygun görüntüleyiciler”:
 - ◆ Kullanıcı yorumu : her farklı doküman tipi için farklı kullanıcı
 - ◆ Geliştirici yorumu : her dokümanın içeriğini gösteren bir metin görüntüleyici

Gereksinimlerin Tamlığı ve Tutarlılığı

- Gereksinimler tam ve birbiriyle tutarlı olarak ifade edilmelidir.
 - ▶ Tamlık : Sistemin beklenen tüm özellikleri tanımlanmalıdır.
 - ▶ Tutarlılık: Sistemin tanımlanan özellikleri arasında çelişkiler bulunmamalıdır.
- Pratikte, doğal dilden kaynaklanan zorluklar sebebiyle, gereksinimleri tam ve tutarlı olarak ifade etmek çok kolay değildir.
- Tanımlanan gereksinimlerin ilgili tüm kişilerce [gözden geçirilmesi](#), tamlığı ve tutarlılığı büyük ölçüde sağlamanın en basit yoludur.

Gereksinimleri Yazmak İçin Öneriler

- Standart bir biçim belirleyerek gereksinimleri tanımlarken kullanın.
- Doğal dili tutarlı olarak kullanın. Zorunlu ve seçimli gereksinimleri farklı kalıplarla ifade edin.
- Gereksinimlerin önemli kısımlarını ayırt etmek için farklı yazı tipi (büyük harf, alt çizme, farklı renk, vb.) kullanın.
- Bilgisayar terimlerini kullanmaktan kaçının.

Gereksinimler ve Tasarım

- Gereksinimler, sistemin “ne” yapacağını tanımlar.
- Tasarım, sistemin tanımlanan gereksinimlerinin “nasıl” gerçekleştirileceğini belirtir.
- Pratikte, gereksinimler ve tasarım her zaman net olarak ayrılamayabilir.
 - ▶ Sistem mimarisi, gereksinimleri yapılandırma için tasarlanır.
 - ▶ Sistem işlevleri, tasarım kısıtlarını belirleyen diğer sistemlerle ilişki içinde gerçekleştiriliyor olabilir.
 - ▶ Müşteri tarafından, sistemin özel bir tasarıma uyması isteniyor olabilir.

Gereksinimlerin türlerine göre ayrı başlıklar altında tanımlanması, bu karışıklığı azaltacaktır.

Doğal Dile İle İlgili Problemler

- Muğlaklık (“Ambiguity”)
 - ▶ Gereksinimler, okuyan herkes tarafından aynı yorumlanacak şekilde yazılmalıdır. Doğal dil muğlak ifadelerle açıktır.
- Aşırı esneklik (“Over-flexibility”)
 - ▶ Bir gereksinim, doğal dil ile çok farklı şekillerde ifade edilebilir.
- Modülerliğin olmayışı (“Lack of modularisation”)
 - ▶ Doğal dilin öğeleri, sistem gereksinimlerini yapısalılaştırmak için yetersiz kalmaktadır.

Bu problemlere rağmen doğal dilin kullanılması, müşteri ve geliştirici arasındaki iletişim açısından önem taşımaktadır.

Doğal Dile Alternatifler

Notation	Description
Structured natural language	This approach depends on defining standard forms or templates to express the requirements specification.
Design description languages	This approach uses a language like a programming language but with more abstract features to specify the requirements by defining an operational model of the system. This approach is not now widely used although it can be useful for interface specifications.
Graphical notations	A graphical language, supplemented by text annotations is used to define the functional requirements for the system. An early example of such a graphical language was SADT. Now, use-case descriptions and sequence diagrams are commonly used.
Mathematical specifications	These are notations based on mathematical concepts such as finite-state machines or sets. These unambiguous specifications reduce the arguments between customer and contractor about system functionality. However, most customers don't understand formal specifications and are reluctant to accept it as a system contract.

Yapısal Doğal Dil – Örnek: Form Esaslı Tanımlama

Insulin Pump/Control Software/SRS/3.3.2

Function Compute insulin dose: Safe sugar level

Description Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.

Inputs Current sugar reading (r2), the previous two readings (r0 and r1)

Source Current sugar reading from sensor. Other readings from memory.

Outputs CompDose – the dose in insulin to be delivered

Destination Main control loop

Action: CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.

Requires Two previous readings so that the rate of change of sugar level can be computed.

Pre-condition The insulin reservoir contains at least the maximum allowed single dose of insulin..

Post-condition r0 is replaced by r1 then r1 is replaced by r2

Side-effects None

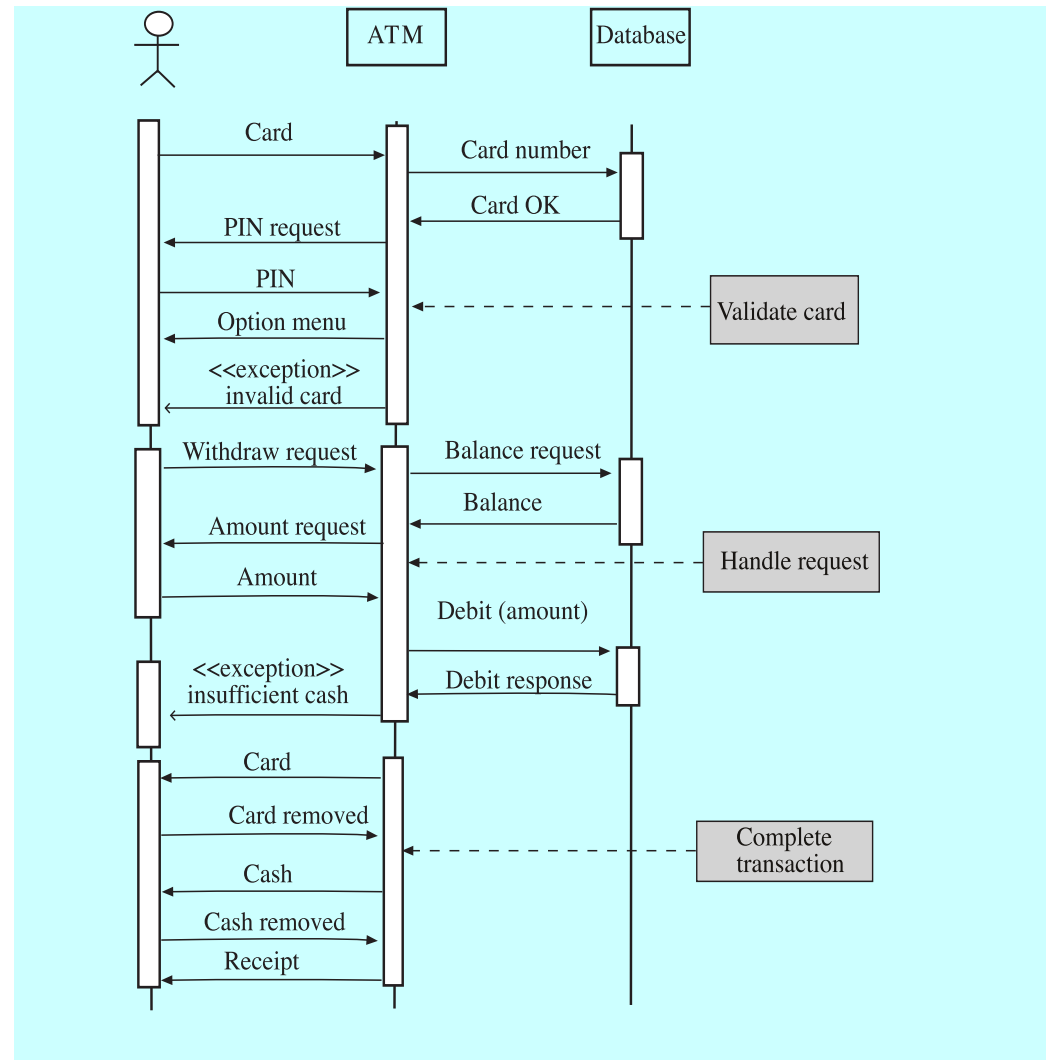
Arayüz (“Interface”) Tanımlama

- Geliştirilen birçok sistem, diğer sistemlerle birlikte çalışmak zorundadır. Birlikte çalışmayı sağlayacak arayüzler de gereksinimlerin bir parçası olarak tanımlanmalıdır.
- Tanımlanabilecek arayüz türleri:
 - ▶ Yordam arayüzleri
 - ▶ Değiş-tokuş edilecek veri yapılarının arayüzleri
 - ▶ Veri gösterimlerine ilişkin arayüzler
- Formal gösterimler, arayüz tanımlamaları için daha uygundur.

Arayüz Tanımlama – Örnek: Java PDL Arayüz Tanımı

```
interface PrintServer {  
  
    // defines an abstract printer server  
    // requires:      interface Printer, interface PrintDoc  
    // provides: initialize, print, displayPrintQueue, cancelPrintJob, switchPrinter  
  
    void initialize ( Printer p ) ;  
    void print ( Printer p, PrintDoc d ) ;  
    void displayPrintQueue ( Printer p ) ;  
    void cancelPrintJob (Printer p, PrintDoc d) ;  
    void switchPrinter (Printer p1, Printer p2, PrintDoc d) ;  
} //PrintServer
```

Grafik Gösterim – Örnek: UML Ardıl İşlem (“Sequence”) Diyagramı



Matematiksel Tanımlama – Örnek: Z Gösterimi

■ <http://archive.comlab.ox.ac.uk/z.html>

- *The Z notation is a formal specification notation based on set theory and predicate calculus.*

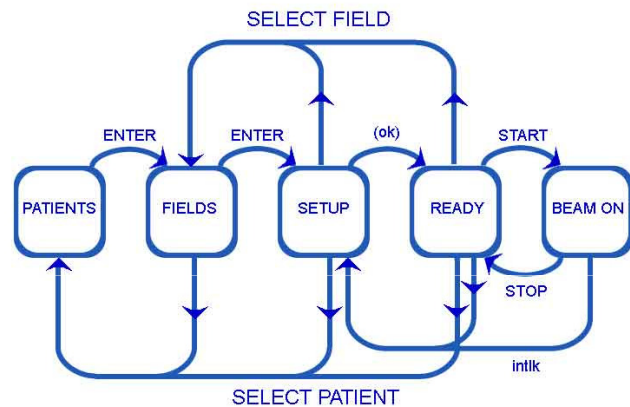


Figure 6.6: Therapy control cascade: state transition diagram
Graphic courtesy of kimberlybatteau.com

STATE ::= patients | fields | setup | ready | beam_on

EVENT ::= select_patient | select_field | enter | start | stop | ok | intlk

FSM == (STATE × EVENT) → STATE

no_change, transitions, control: FSM

control = no_change ⊗ transitions

no_change = { s: STATE; e: EVENT • (s, e) → s }

transitions = { (patients, enter) → fields,

(fields, select_patient) → patients, (fields, enter) → setup,

(setup, select_patient) → patients, (setup, select_field) → fields, (setup, ok) → ready,

(ready, select_patient) → patients, (ready, select_field) → fields, (ready, start) → beam_on, (ready, intlk) → setup,

(beam_on, stop) → ready, (beam_on, intlk) → setup }

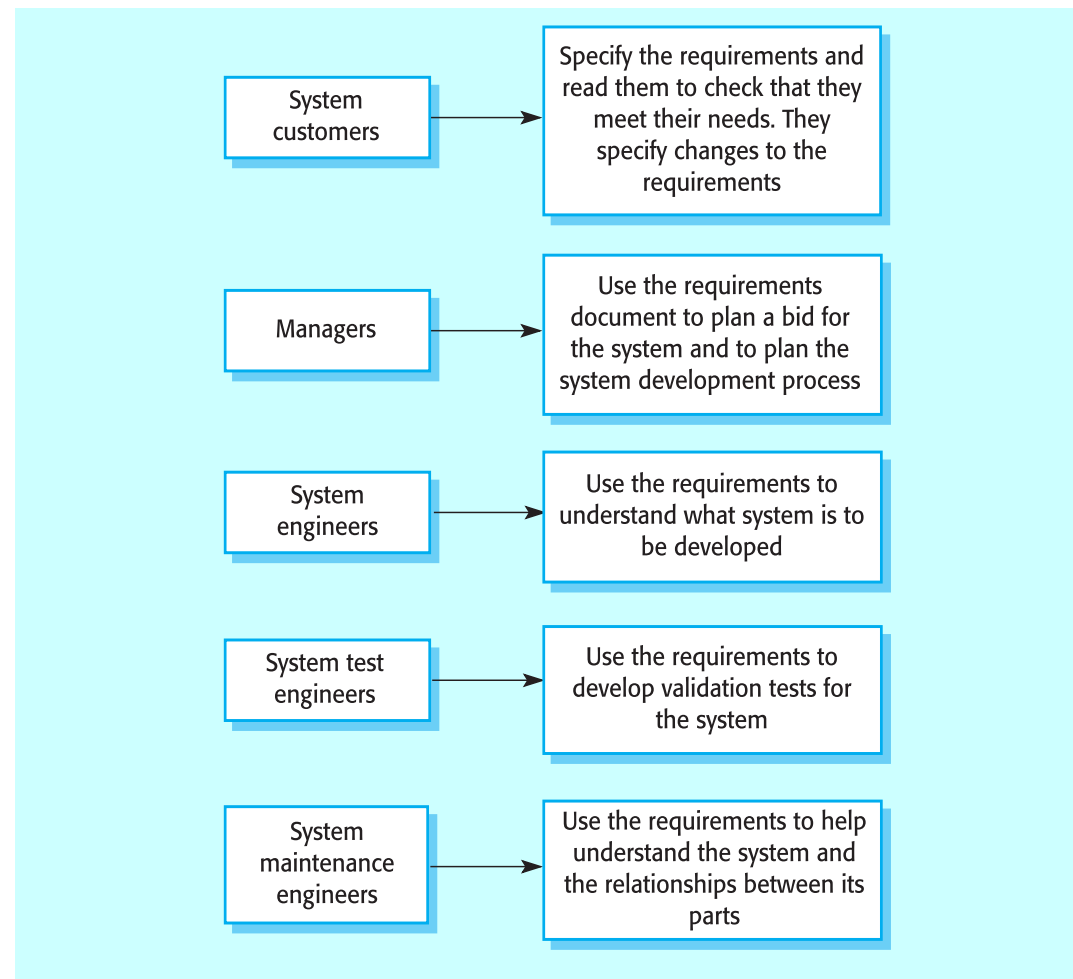
Gereksinim Belgesi

- Sistemin sağlaması beklenen özelliklerin resmi tanımıdır.
- Hem kullanıcı gereksinimlerini, hem de sistem gereksinimlerini içermesi beklenir.
 - ▶ Bazı modeller bu ikisi için ayrı belgelerin oluşturmasını önermektedir.
- Mümkün olduğunca sistemin “ne yapacağını” tanımlamalı, “nasıl yapacağı” detayına girmemelidir.

Gereksinim Belgesi Tipik İçeriği

- Önsöz
- Giriş
- Tanımlar
- Kullanıcı gereksinimleri
- Sistem mimarisi
- Sistem gereksinimleri
- Sistem modelleri
- Sistem gelişimi
- Ekler
- Endeks

Gereksinim Belgesinin Kullanıcıları



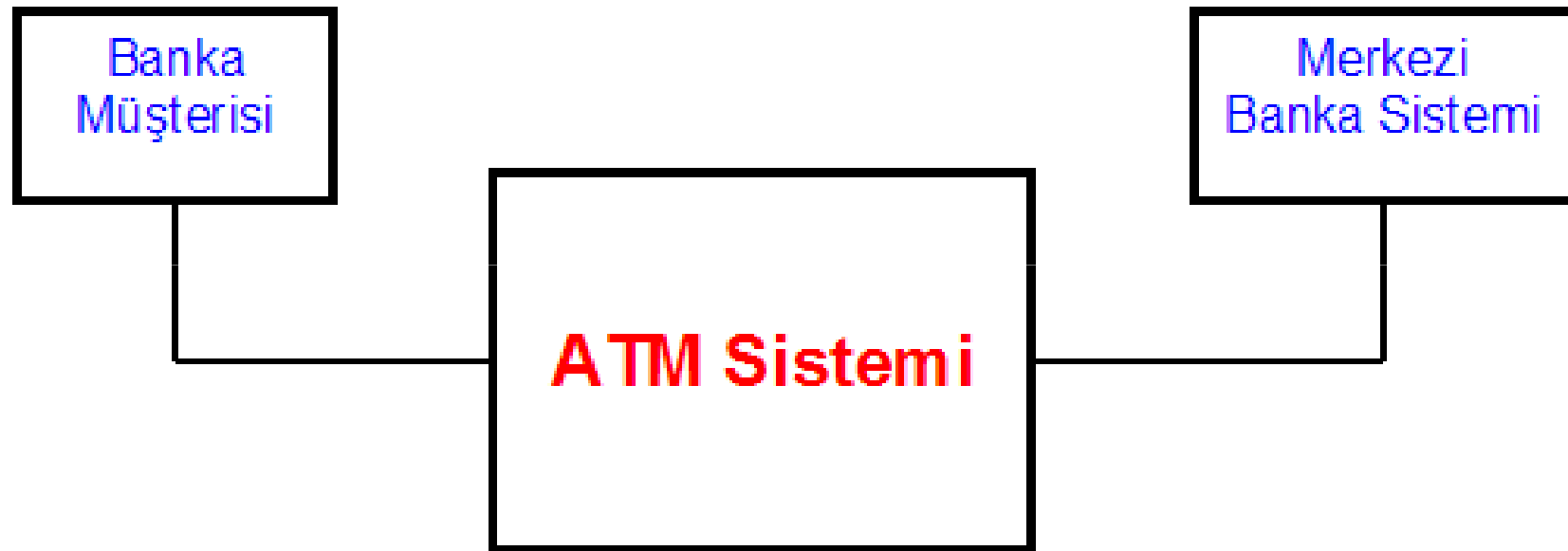
IEEE Gereksinim Belgesi Standardı

- IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
- Gereksinim belgesi için, her sistem için özelleştirilebilecek genel bir yapı sunar.
 - ▶ “Introduction”
 - ▶ “General description”
 - ▶ “Specific requirements”
 - ▶ “Appendices”
 - ▶ “Index”

Örnek: ATM Uygulaması

- Bir bankanın ATM cihazı için yazılım geliştirilecektir. ATM, banka kartı olan müşterilerin hesaplarından para çekmelerine, hesaplarına para yatırmalarına ve hesapları arasında para transferi yapmalarına olanak sağlayacaktır. ATM, banka müşterisi ve hesapları ile ilgili bilgileri, gerektiğinde merkezi banka sisteminden alacaktır. Banka sistemi ayrıca her günün sonunda, ATM'den günlük işlemlerin bir özetini isteyecektir.

ATM Uygulaması – Kapsam



Bağlam (“context”) diyagramı

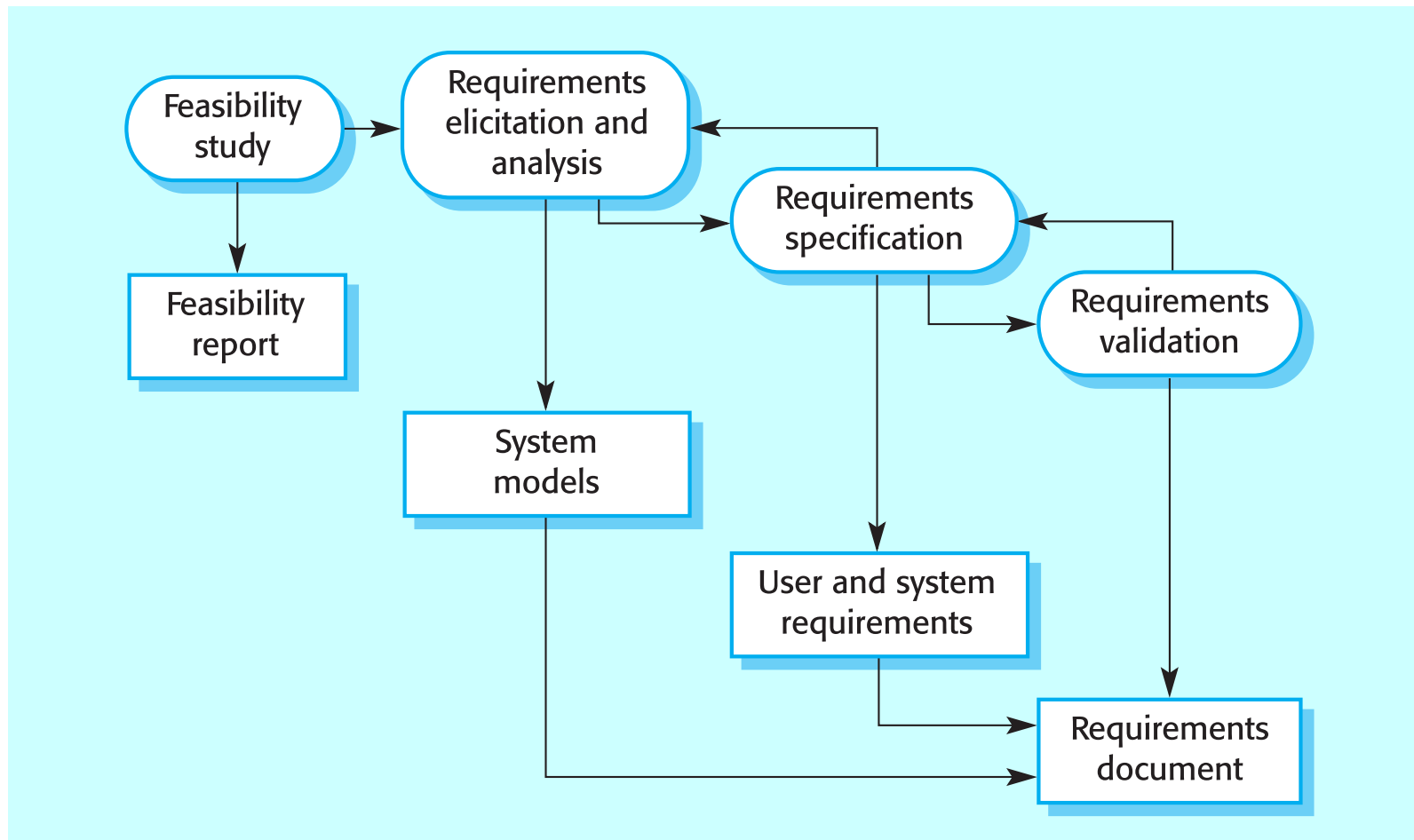


Gereksinim Mühendisliği Süreci

Gereksinim Mühendisliği Süreci (1)

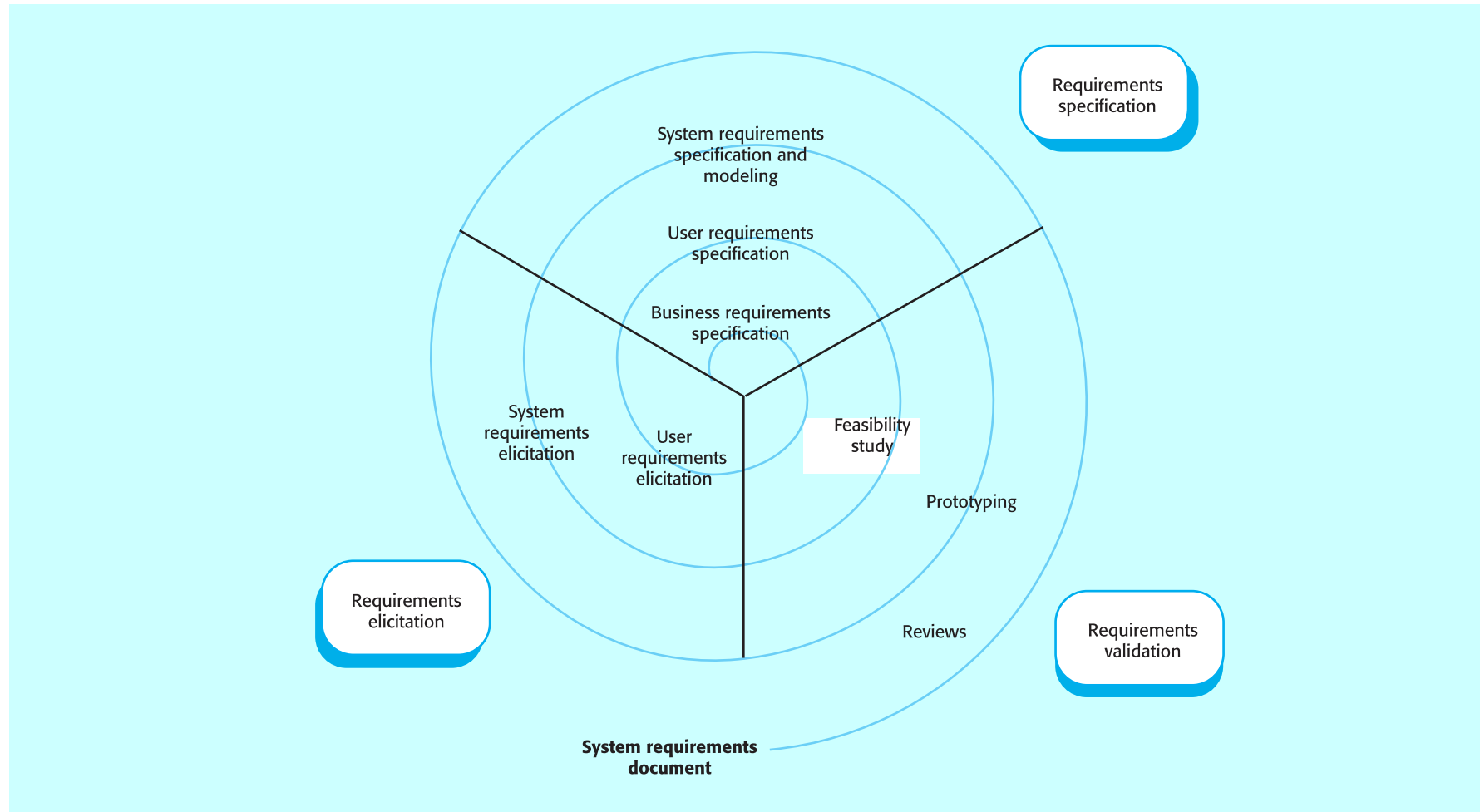
- Temel olarak aşağıdaki etkinlikleri içerir:
 - ▶ Gereksinim çıkarma (“Requirements elicitation”)
 - ▶ Gereksinim analizi (“Requirements analysis”)
 - ▶ Gereksinim geçerleme (“Requirements validation”)
 - ▶ Gereksinim yönetimi (“Requirements management”)

Gereksinim Mühendisliği Süreci (2)



Gereksinim Mühendisliği Süreci (3)

– Etkinliklerin Gelişimi



Olurluk Çalışmaları (“Feasibility Studies”)

- Olurluk çalışmasının amacı, sistemin gerçekleştirilip gerçekleştirilmeyeceğine karar vermektir.
- Bilgi toplamaya, değerlendirmeye ve raporlamaya dayanır.
 - ▶ Sistem kurumsal hedeflere hizmet edecek mi?
 - ▶ Sistem mevcut teknoloji ve bütçe sınırları içinde gerçekleştirilecek mi?
 - ▶ Sistem mevcut diğer sistemlere entegre edilebilecek mi?

Gereksinim Çıkarma ve Analizi (1)

- Uygulama (iş) alanını anlamayı ve sistemin bu alanı ne kapsamda destekleyeceğini bulmayı gerektirir.
 - ▶ İş alanında yürütülen işlevler nelerdir ve nasıl ilişkilidir?
 - ▶ Sistem bu işlevleri ne kapsamda ve hangi noktalarda destekleyebilir?
 - ▶ Sistemin destekleyeceği iş kapsamı için gereksinimler ve kısıtlar nelerdir?
- Son kullanıcılar, yöneticiler, alan uzmanları ve geliştiriciler; gereksinim çıkarma ve analiz etkinliklerine dahil olur. Tüm bu kişiler *sistemin paydaşları* (“*stakeholders*”) olarak isimlendirilir.

Gereksinim Çıkarma ve Analizi (2)

■ Yapılan hatanın düzeltilme maliyeti çok yüksek

- ▶ Bakım aşamasında 20 kat fazla
- ▶ Bulunan hataların %40'ı gereksinimlerden kaynaklanıyor

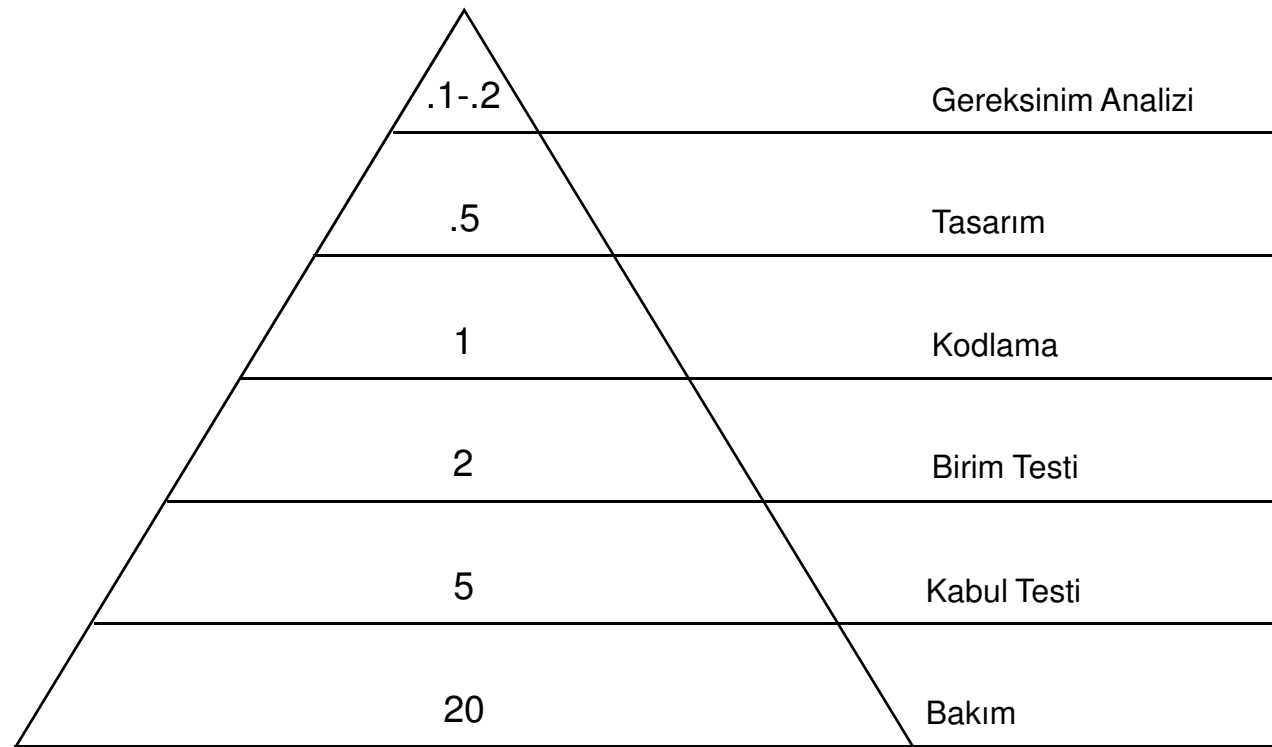
■ “Doğru” sistemi oluşturmak için

- ▶ Kurumsal ortam nedir ?
- ▶ Müşteri ne istiyor ?
- ▶ Bağlı bulunduğu sistemler var mı ?

■ Gereksinim tanımı iletişime esas

- ▶ Müşteri – teknik ekip
- ▶ Analiz uzmanları – tasarım uzmanları

Gereksinim Analizi ve Hata Düzeltme Maliyeti



Karşılaştırmalı Hata Düzeltme Maliyeti Oranı

Referans: Managing Software Requirements – A Unified Approach, Leffingwell & Widrig, 2000

Gereksinim Çıkarma ve Analizi – Zorluklar

- Yazılım geliştirme çalışmalarının ön aşamaları
 - ▶ Kimse birbirini tanımıyor, sistemi bilmiyor
 - ▶ Yöntem, teknoloji, vb. yeni olabilir
- Kullanıcı odaklı problemler yaşanabilir.
 - ▶ Kullanıcılar ne istediklerini bilmeyebilir.
 - ▶ Kullanıcılar isteklerini kendi terimleriyle ifade edebilir.
 - ▶ Farklı kullanıcıların çelişen istekleri olabilir.
- Farklı grupların beraber çalışmasını gerektiriyor.
 - ▶ Kullanıcı grupları, analiz uzmanları, mimari/tasarım uzmanları
- Kurumsal ve politik etkenler sistem gereksinimlerini etkileyebilir.
- Analiz boyunca gereksinimler değişebilir; yeni kullanıcılar çıkabilir ve iş ortamı değişebilir.

Gereksinim Geçerleme

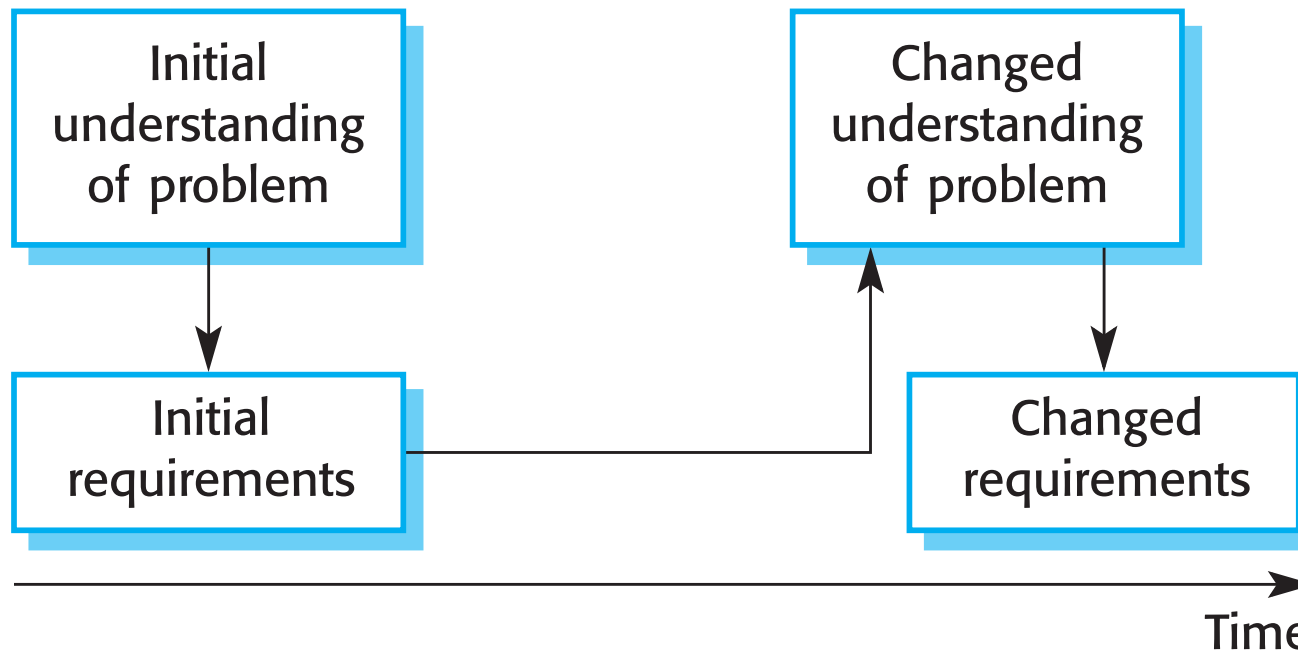
- Gereksinimlerin müşterinin istediği sistemi tanımladığını güvence altına almayı hedefler.
- Gereksinimlerden kaynaklanan bir hatayı sonradan düzeltmenin maliyeti yüksek olduğundan, geçerleme büyük önem taşır.
- Gereksinim geçerleme teknikleri:
 - ▶ Gözden geçirme – gereksinimlerin sistematik ve paydaşlara dayalı analizi
 - ▶ Prototip oluşturma – sistemin çalışan bir taslağı üzerinden kontrol
 - ▶ Test durumu geliştirme – sistemin test edilebilirliğini kontrol amacıyla gereksinimler için test durumu geliştirme

Gözden Geçirme

- Gereksinimlerinin gözden geçirilmesi, hataların en aza indirilmesi, kalitenin güvencesi ve projenin başarısı açısından son derece önemlidir.
 - ▶ Gereksinim aşaması yazılım geliştirme sürecindeki en zayıf noktadır.
 - ▶ Yapılan hatalar ancak sürecin çok ilerleyen aşamalarında (büyük olasılıkla kabul testlerinde) ortaya çıkacaktır.
 - ▶ Analiz ekibi gerek kendi içinde gerekse sistemin kullanıcıları ile gereksinimleri gözden geçirmelidir.
- Gözden geçirme çalışmalarında hedefimiz, yazılım gereksinimlerinin aşağıdaki özellikleri taşıdığından emin olmaktır:
 - ▶ Tam ve doğru
 - ▶ Anlaşılabilir
 - ▶ Tutarlı
 - ▶ Test edilebilir
 - ▶ Diğer belgelerde tanımlanan iş/kullanıcı/sistem gereksinimlerine izlenebilir

Gereksinim Yönetimi

- Geliştirme boyunca değişen gereksinimlerin yönetilmesi etkinliĒidir.
 - Geliştirme ilerledikçe (ve sisteme ilişkin anlayış iyileştikçe) yeni gereksinimler ortaya çıkabilir.
 - Farklı bakış açılarının gereksinimleri zaman içinde çelişebilir.



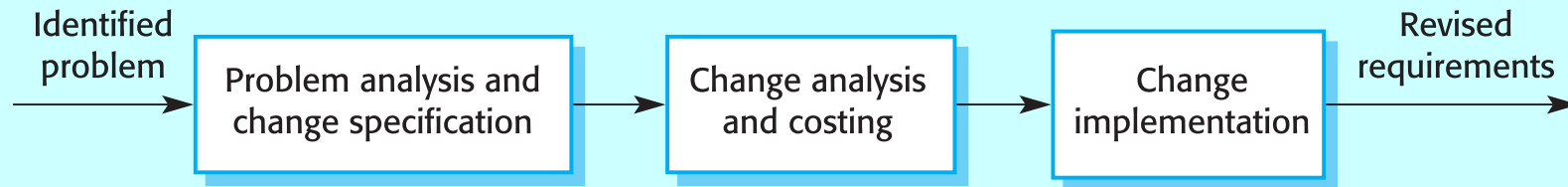
- Gereksinim Belgesinde yer alan her gereksinim, analizde kaynak olarak kullanılan belgelere ve sistem tasarımına izlenebilir olmalıdır.
- İzlenebilirlik aşağıdaki gibi sınıflandırılabilir:
 - ▶ Kaynağa olan izlenebilirlik
 - ◆ Gereksinimlerden, onu oluşturan paydaşlara ve kaynak belgelere
 - ▶ Gereksinimlere olan izlenebilirlik
 - ◆ Gereksinimlerden, bağımlı diğer gereksinimlere
 - ▶ Tasarıma olan izlenebilirlik
 - ◆ Gereksinimlerden tasarıma
- Gereksinimler ve diğer belgeler arasındaki ilişki bir matrisle tanımlanır.
 - ▶ Gözden geçirme sırasında bu matristeki bilginin doğruluğu, tamlığı ve tutarlılığı kontrol edilir.
 - ▶ İzlenebilirlik çift yönlü olarak sağlanmalıdır (geliştirme boyunca ileri ve geri).

İzlenebilirlik Matrisi – Örnek: Gereksinimlere İzlenebilirlik

Req. id	1.1	1.2	1.3	2.1	2.2	2.3	3.1	3.2
1.1		D	R					
1.2			D			D		D
1.3	R			R				
2.1			R		D			D
2.2								D
2.3		R		D				
3.1								R
3.2							R	

D: bağımlı (“dependent”)
R: ilişkili (“relational”)

Gereksinim Yönetimi Adımları



Genellikle bir CASE aracı ile desteklenir.

CMMI – Gereksinim Geliştirme (“Requirements Development”) Süreç Alanı (ML3)

- SG 1 Müşteri gereksinimlerini geliştir
 - ▶ SP 1.1 İhtiyacı çıkar
 - ▶ SP 1.2 Müşteri gereksinimlerini geliştir

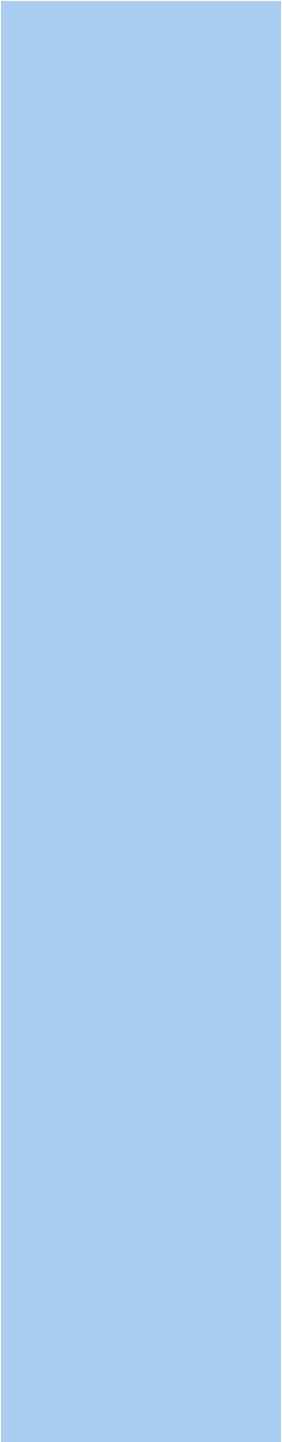
- SG 2 Ürün gereksinimlerini geliştir
 - ▶ SP 2.1 Ürün ve ürün bileşen gereksinimlerini oluştur
 - ▶ SP 2.2 Ürün bileşen gereksinimlerini ata
 - ▶ SP 2.3 Arayüz gereksinimlerini belirle

- SG 3 Gereksinimleri analiz et ve onayla
 - ▶ SP 3.1 İşlevsel kavramları ve senaryoları oluştur
 - ▶ SP 3.2 Beklenen işlevselliğin tanımını oluştur
 - ▶ SP 3.3 Gereksinimleri analiz et
 - ▶ SP 3.4 Gereksinimleri dengeyi sağlamak için analiz et
 - ▶ SP 3.5 Gereksinimleri onayla

CMMI – Gereksinim Yönetimi (“Requirements Management”) Süreç Alanı (ML2)

■ SG 1 Gereksinimleri yönet

- ▶ SP 1.1 Gereksinimlerin anlaşılmasını sağla
- ▶ SP 1.2 Gereksinimlere taahhütü sağla
- ▶ SP 1.3 Gereksinim değişikliklerini yönet
- ▶ SP 1.4 Gereksinimlerin çift-yönlü izlenebilirliğini sürdür
- ▶ SP 1.5 Proje adımları ve gereksinimler arasındaki tutarsızlıkları belirle



İş Gereksinimleri Analizi – Gereksinim Analizi İlişkisi

Yazılım Yaşam Döngüsü

- Girdi: İş gereksinimleri (iş alanı bilgisini gerektirir)
-

- Yazılım ürününün geliştirilmesi ve yönetilmesi için izlenmesi gereken adımların bütünüdür

- ▶ Analiz, tasarım, kodlama, test, ...
- ▶ Proje yönetimi, kalite yönetimi, yapılandırma yönetimi, ...

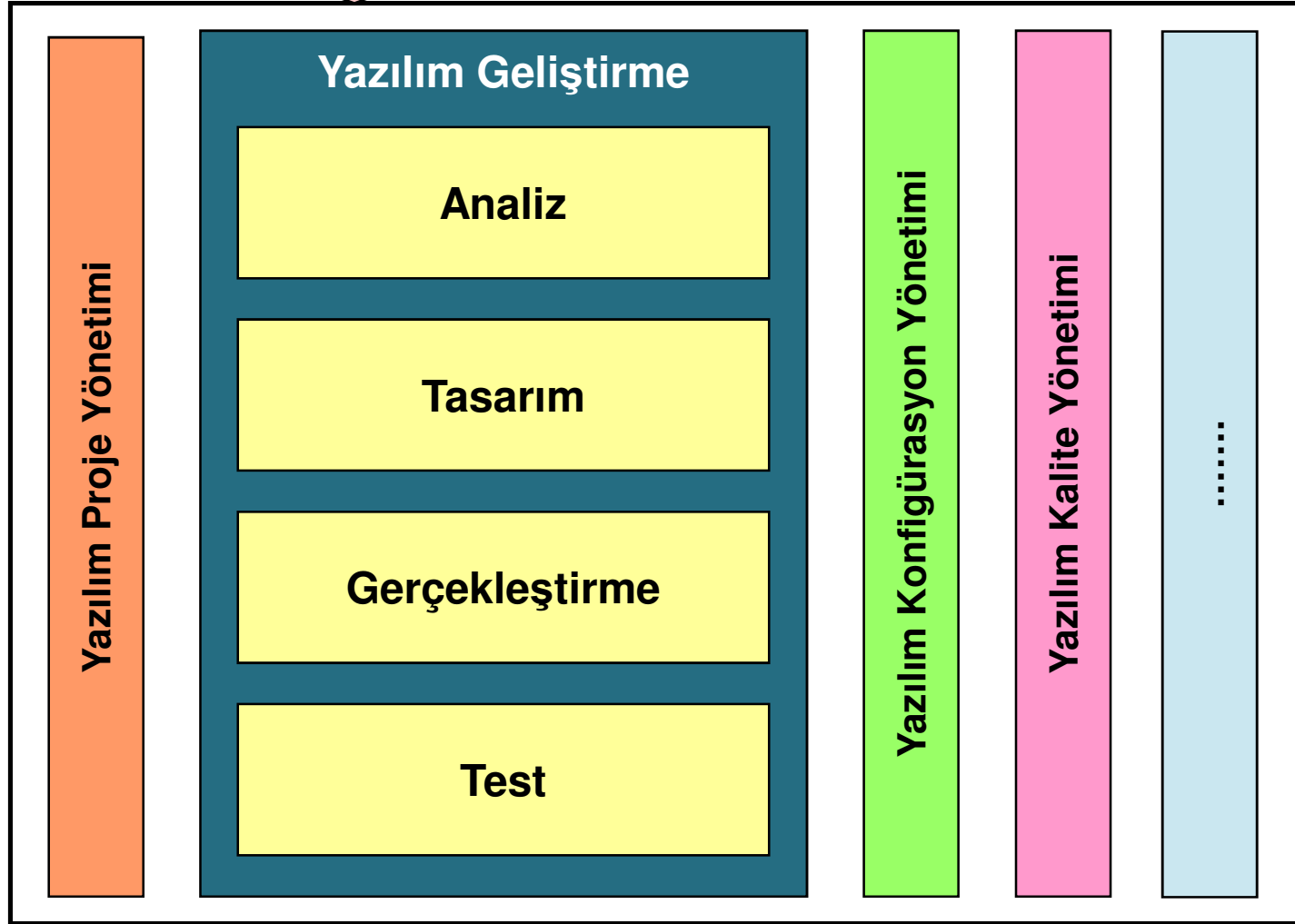
- Yazılım ürününe mühendislik etkinliklerinin uygulanmasına olanak sağlar

- ▶ Örnek: ISO/IEC 12207 Yazılım Yaşam Döngüsü Süreçleri
-

- Çıktı: Yazılım sistemi (çözüm alanı bilgisini gerektirir)

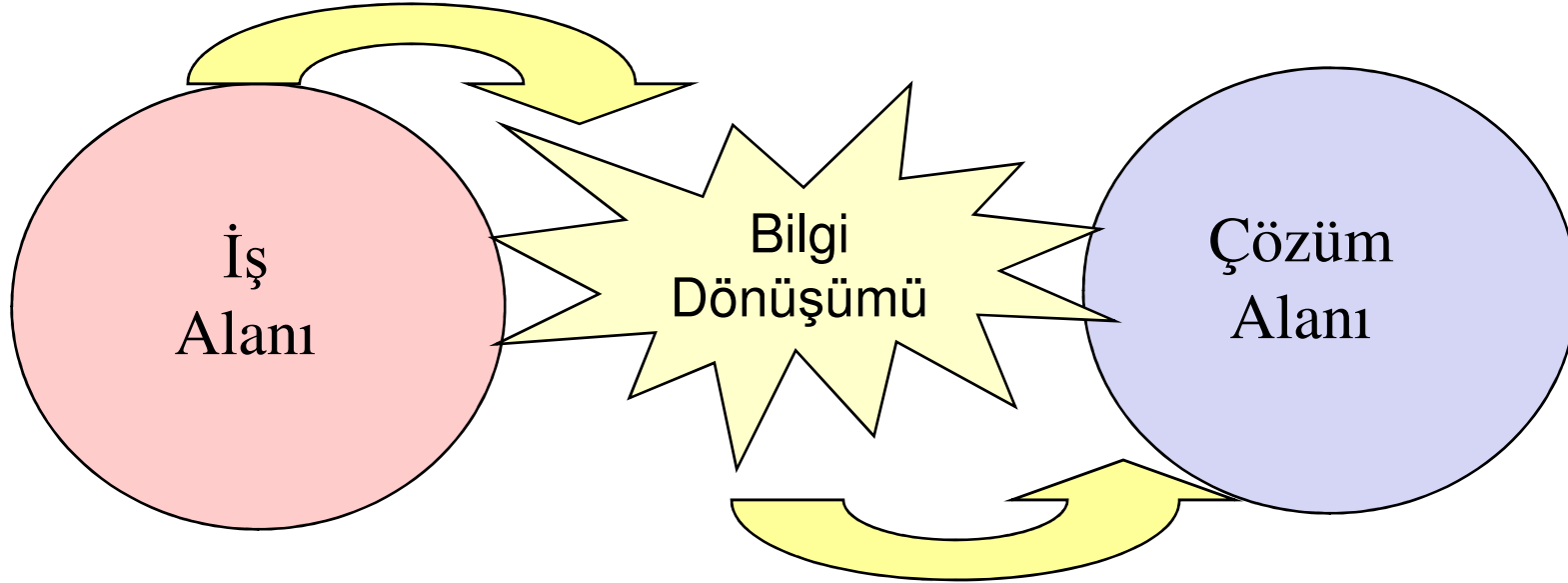
Müşteri İhtiyaçları

Yazılım Yaşam Döngüsü



Yazılım Ürünü

İş Gereksinimleri Analizi



*Yazılım sisteminin başarısında;
iş alanına ilişkin bilgi, en az çözüm alanına
ilişkin bilgi kadar önemlidir*

İş Gereksinimleri Analizi

- Amaç: İş alanına (“business domain”) ilişkin kavramları, süreçleri, ortamı ve kullanıcı gereksinimlerini anlamak
 - ▶ Yazılım sisteminin içinde çalışacağı iş ortamı belirlenir
 - ▶ İş ihtiyaçları tanımlanır
 - ▶ İhtiyaç tanımında fikir birliğine varılır
 - ▶ Yeni sistemden etkilenecek kişiler ve kullanıcılar tespit edilir
 - ▶ Çözüm sisteminin çerçevesi çizilir
 - ▶ Çözümü etkileyebilecek kısıtlar ortaya koyulur

Süreç (“Process”) ve İş Süreci (“Business Process”)

■ Süreç (“process”) [Davenport 1993]:

- ▶ “a specific ordering of work activities across time and place, with a beginning, an end, and clearly defined inputs and outputs -- structure for action”

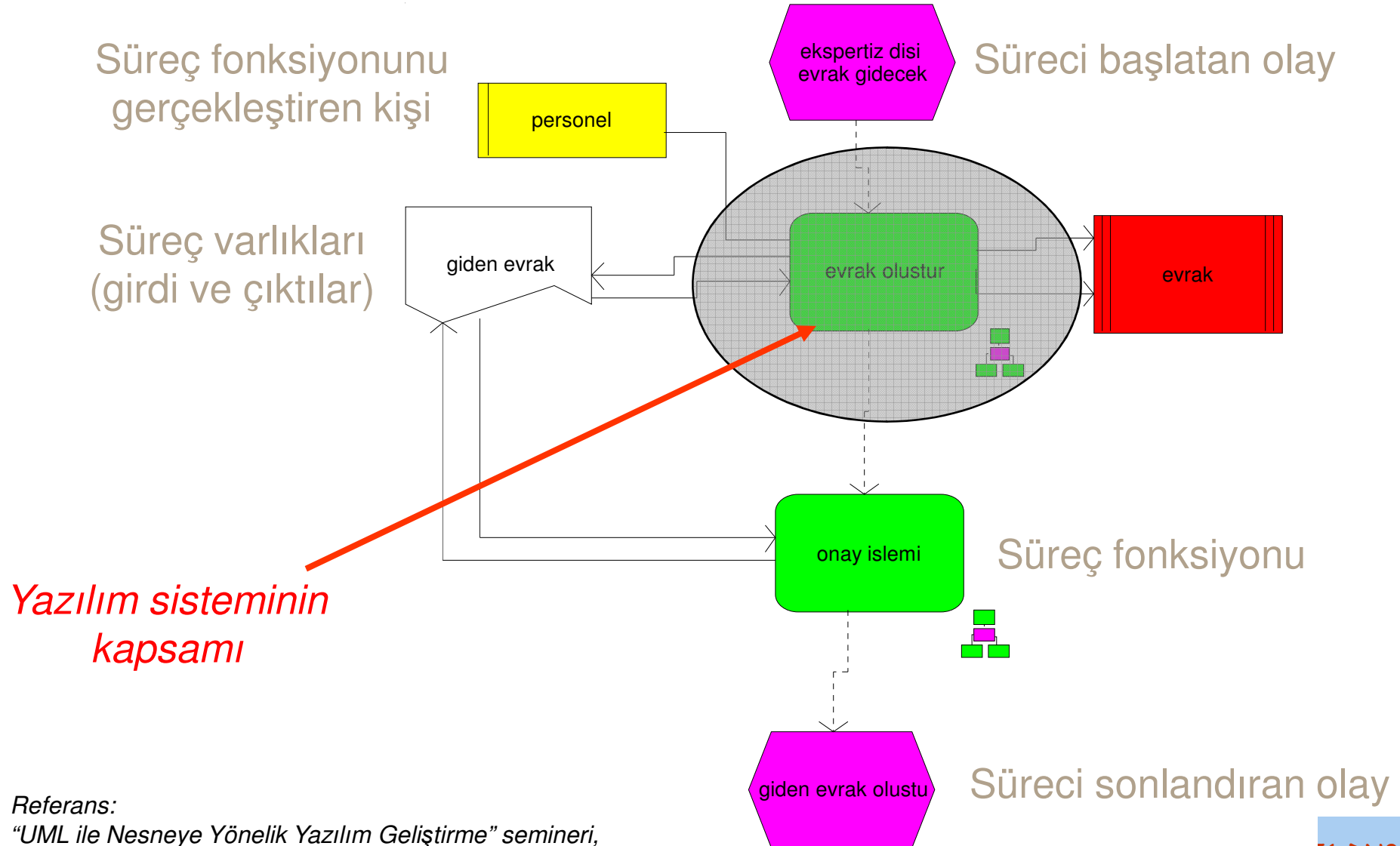
■ İş süreci (“business process”)

- ▶ “a process that defines how an organization achieves its purpose including vision and goals”

İş Süreçlerinin Modellenmesi

- Bilgi sistemi/bilgi teknolojisi ortamları için yaygın olarak kullanılan bir iş analizi yöntemidir
- Amaç:
 - ▶ Kuruluşa ilişkin dinamikleri anlamak
 - ▶ Müşteri, son kullanıcı ve geliştiricilerin, kuruluş süreçlerinden aynı şeyleri anladıklarını garanti etmek
- Uygulama ortamının karmaşık, çok boyutlu ve çok kullanıcıli olduğu durumlarda, getirileri maliyetini daha çok karşılar [Yourdon 2000]

İş Süreçlerinin Modellenmesi - Örnek



Referans:

"UML ile Nesneye Yönelik Yazılım Geliştirme" semineri,
Bilgi Grubu Ltd., 2006

İş Süreçlerinin Modellenmesi – Kazançlar

- İş alanına daha geniş bir bakış açısı getirir.
- Mevcut süreçte aksayan yönleri görmek ve iyileştirmek fırsatını doğurur.
- İş alanındaki kişilerle çözüm alanındaki geliştiriciler arasında ortak bir dil oluşturur.
 - ▶ Müşterinin gereksinim çıkarma sürecine katılımı artar.
 - ▶ Kapsama ilişkin daha sonra yaşanabilecek güçlükler azalır.

İş Süreçleri ve Gereksinim Analizi (1)

- İş süreçlerinin tanımlı olduğu durumlarda aşağıdakiler de tanımlıdır:
 - ▶ Sistemin hangi iş süreci adımlarında kullanılacağı
 - ▶ Sistemin girdileri
 - ▶ Sistemden beklenen çıktılar/sonuçlar
 - ▶ Sistemi kimlerin, hangi amaçla kullanacakları

- Gereksinim analizinin hedefi:
 - ▶ İş süreçlerinin tanımladığı bağlamda detaylı yazılım özelliklerini belirlemek

İş Süreçleri ve Gereksinim Analizi (2)

- İş süreçlerinin tanımsız olduğu veya sadece kişilerin uzmanlıklarında gizli olduğu durumlarda:
 - ▶ Gereksinim analizi süreç modellemesi etkinliklerini de içermeye başlar
 - ▶ Yazılım geliştiriciler bu konuda en yetkin kişiler değildir
 - ▶ Bütçe ve zaman buna göre ayarlanmamıştır

İş Süreçleri ve Gereksinim Analizi (3)

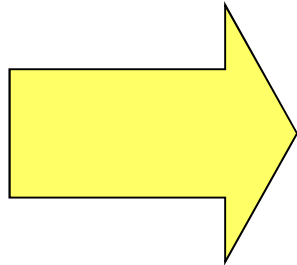
- Geliştirilecek yazılım ürününden beklenen özellikleri *anlamak* ve *tanımlamak*

- İş süreçleri/kuralları

Yapılan iş ne?

İşe ilişkin kısıtlar ne?

Nasıl bir ürün?



Gereksinim tanımı

Anlaşılır, tam, doğru

İletişime esas

Geliştirme sürecine esas

Gereksinim Analizi – Kazançlar

- Artan müşteri memnuniyeti
 - ▶ Müşteri/kullanıcılar çalışmalara katılıyor
 - ▶ İstekler tam ve doğru olarak tanımlanıyor
- Artan yazılım kalitesi
 - ▶ Gereksinimler doğru ve tam olarak tanımlanıyor
 - ▶ Gereksinimleri tüm yazılım ekibi biliyor
 - ▶ Gereksinim değişiklikleri en aza indirilebilecek
- Etkin proje yönetimi
 - ▶ Daha doğru tahminleme (takvim ve bütçe)
 - ▶ Daha kolay izleme (gereksinimler esaslı)
 - ▶ Daha düzgün iş atamaları (gereksinimler esaslı)

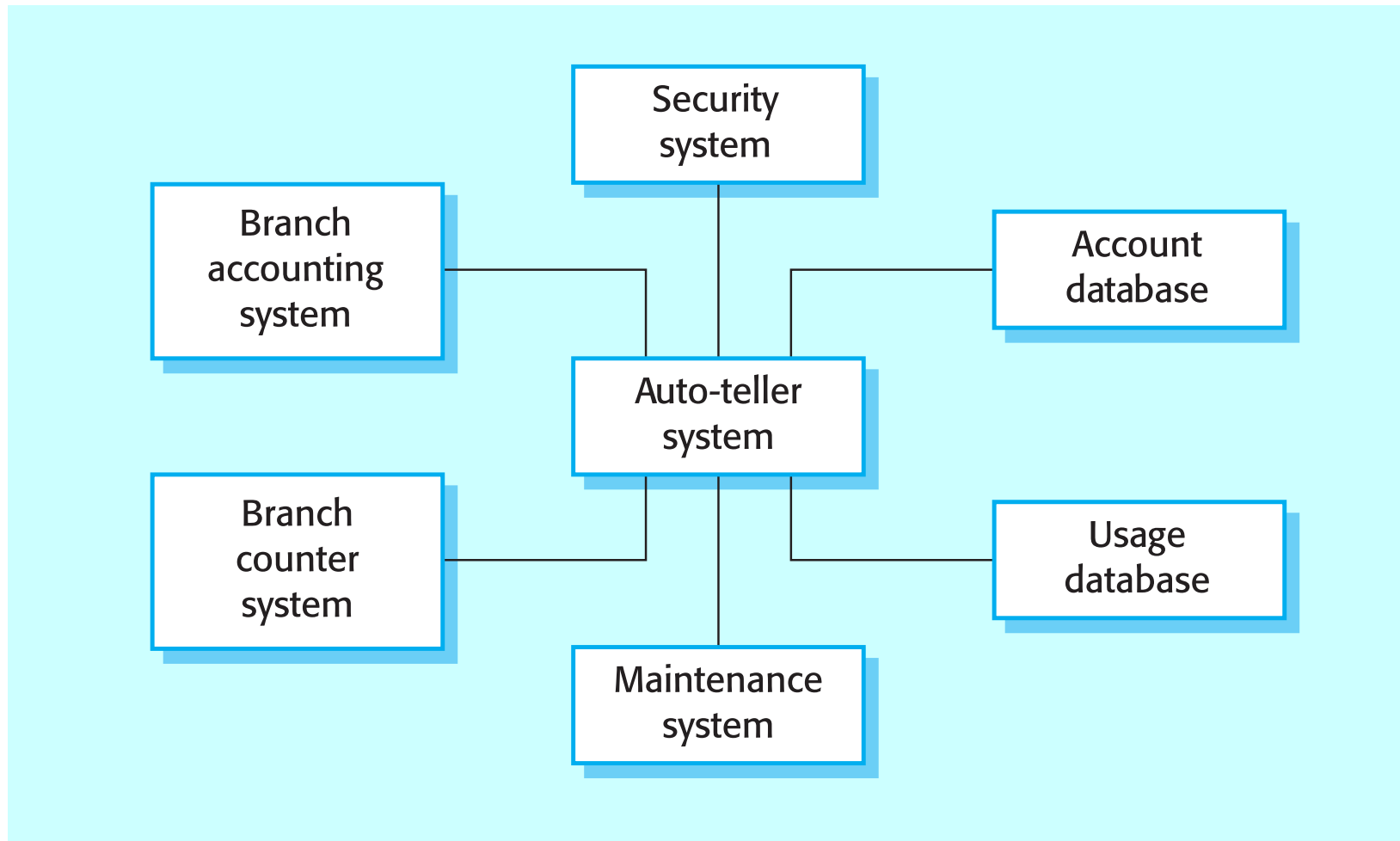


Sistem Modelleri

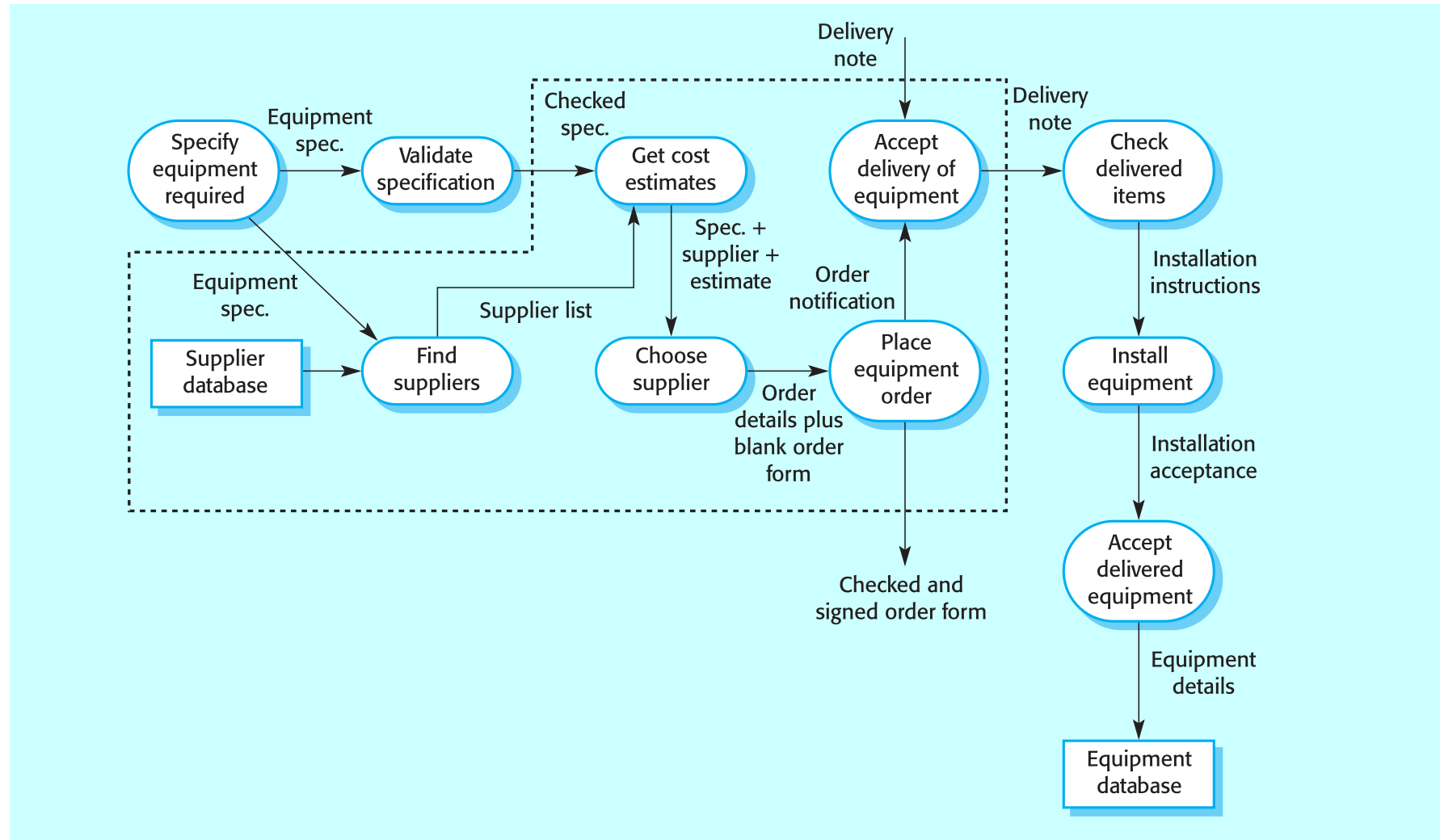
Sistem Modelleme

- Sistem modelleme; gereksinim çıkarma ve analizi aşamaları boyunca sistemin işlevselliğini anlamak için, müşteri ve teknik ekip ile iletişime esas olarak gerçekleştirilir.
- Model, sistemin soyut bir gösterimidir. Tamamlayıcı modeller, sistem hakkında farklı bilgileri ortaya çıkarır.
 - ▶ **Bağlam (“context”) modeli**, sistemin kendi ortamındaki konumunu ve diğer sistem ve süreçlerle olan etkileşimini tanımlar.
 - ▶ **Süreç (“process”) modeli**, sistemin destekleyeceği işleri ve iş akışlarını tanımlar.
 - ▶ **Davranış (“behavioral”) modeli**, sistemin davranışını tanımlar.
 - ◆ Veri-akış (“data flow”) modeli, sistem içindeki verinin akışını tanımlar.
 - ◆ Durum-makinesi (“state-machine”) modeli, sistemin davranışını iç ve dış olaylara gösterdiği tepkilerle tanımlar.
 - ▶ **Veri (“data”) modeli**, sistemin işlediği verinin anlamsal biçimini tanımlar.
 - ▶ **Nesne (“object”) modeli**, sistemdeki veriyi ve nasıl işlemlendiğini tanımlar.
 - ◆ Nesne kalıtım (“object inheritance”) modeli
 - ◆ Nesne bileşen (“object aggregation”) modeli
 - ◆ Nesne davranış (“object behavior”) modeli

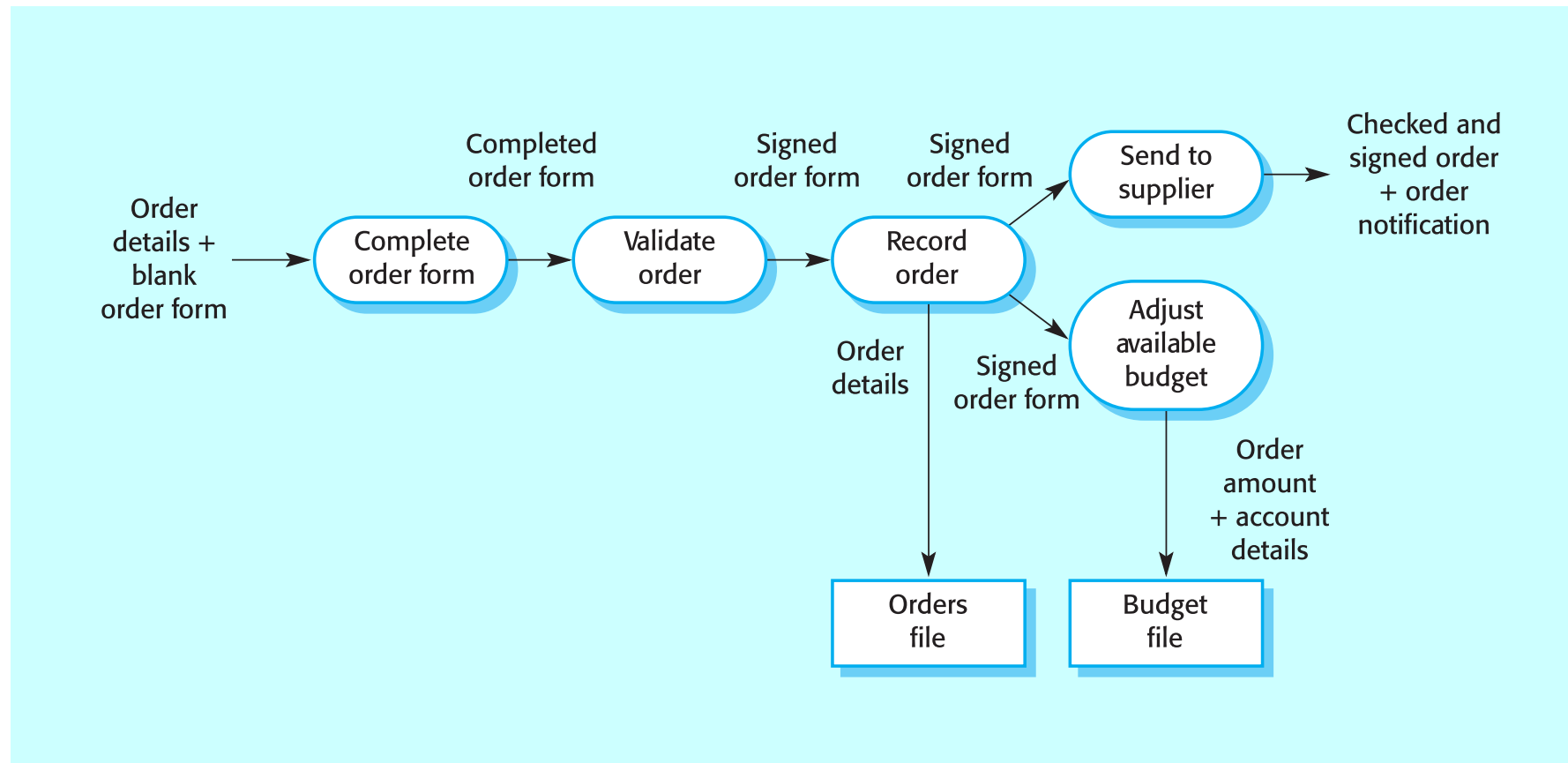
Bağlam (“Context”) Modeli – Örnek: ATM Sistemi



Süreç (“Process”) Modeli – Örnek: Donanım Satınalma Süreci Modeli



Veri-Akış (“Data-Flow”) Modeli – Örnek: Sipariş İşleme Veri-Akış Modeli



Veri-Akış (“Data-Flow”) Modeli – Bir Başka Örnek: “Data Flow Diagram (DFD)”

Figure 9-1: Data Flow Diagram (Gane-Sarson Notation)

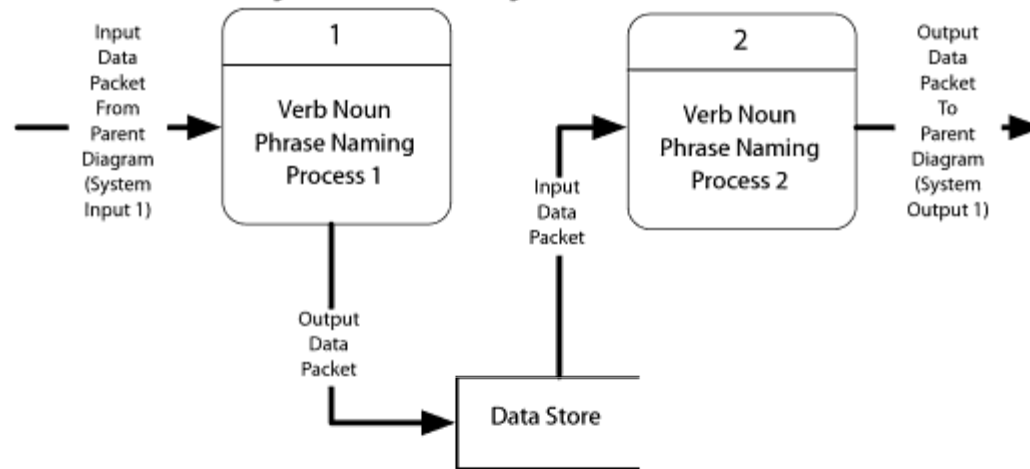
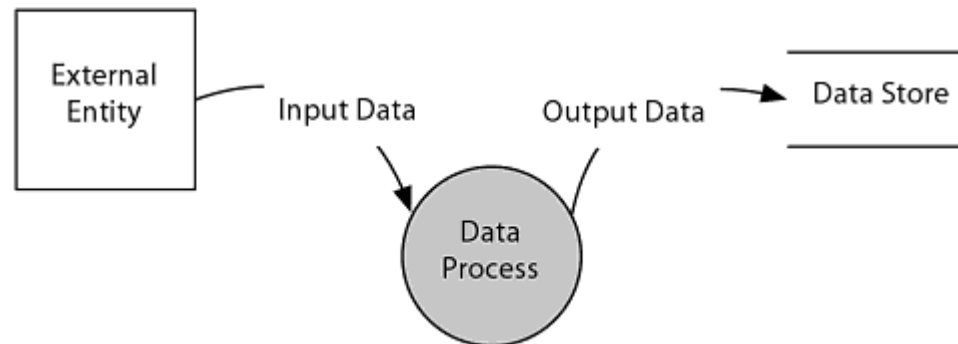
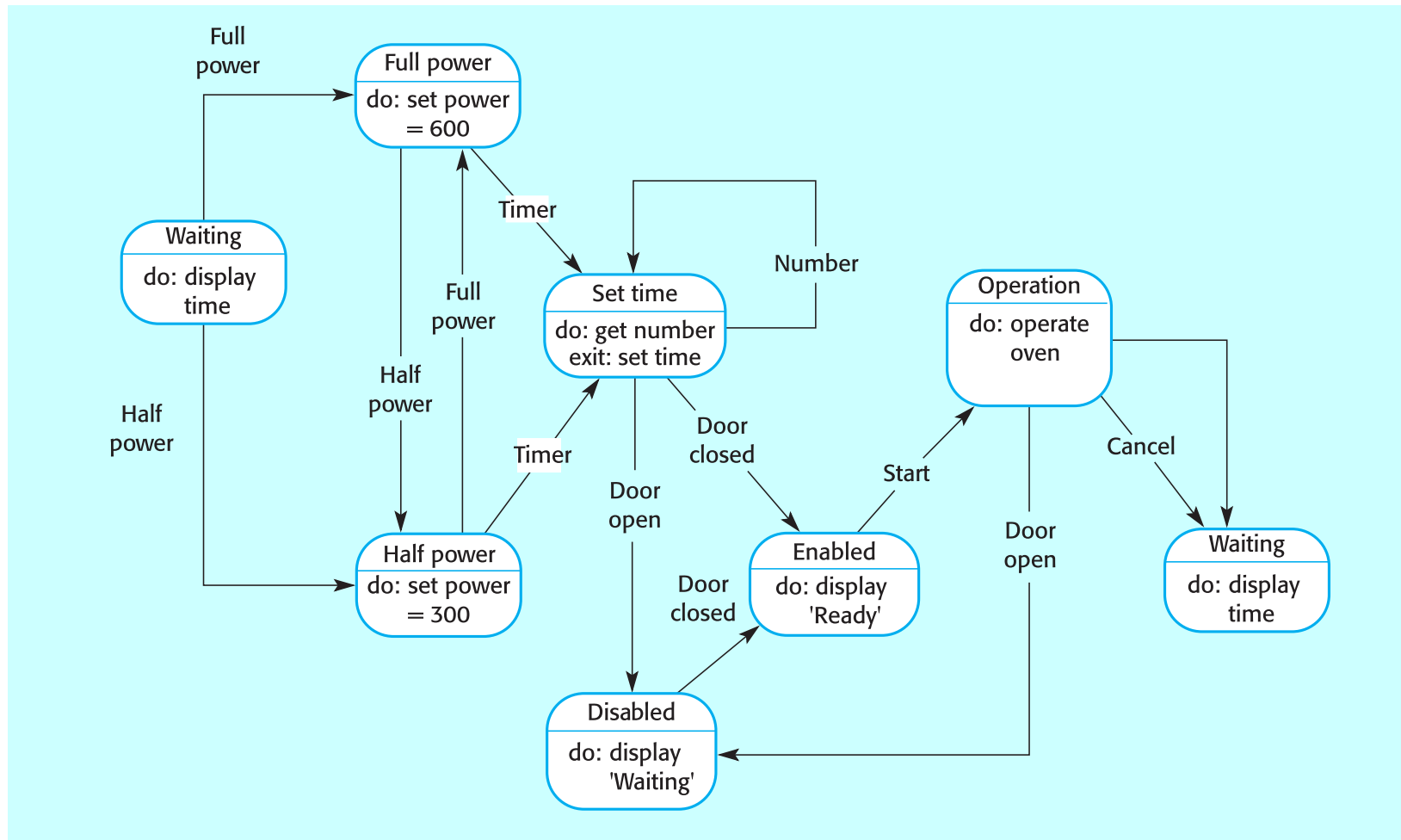


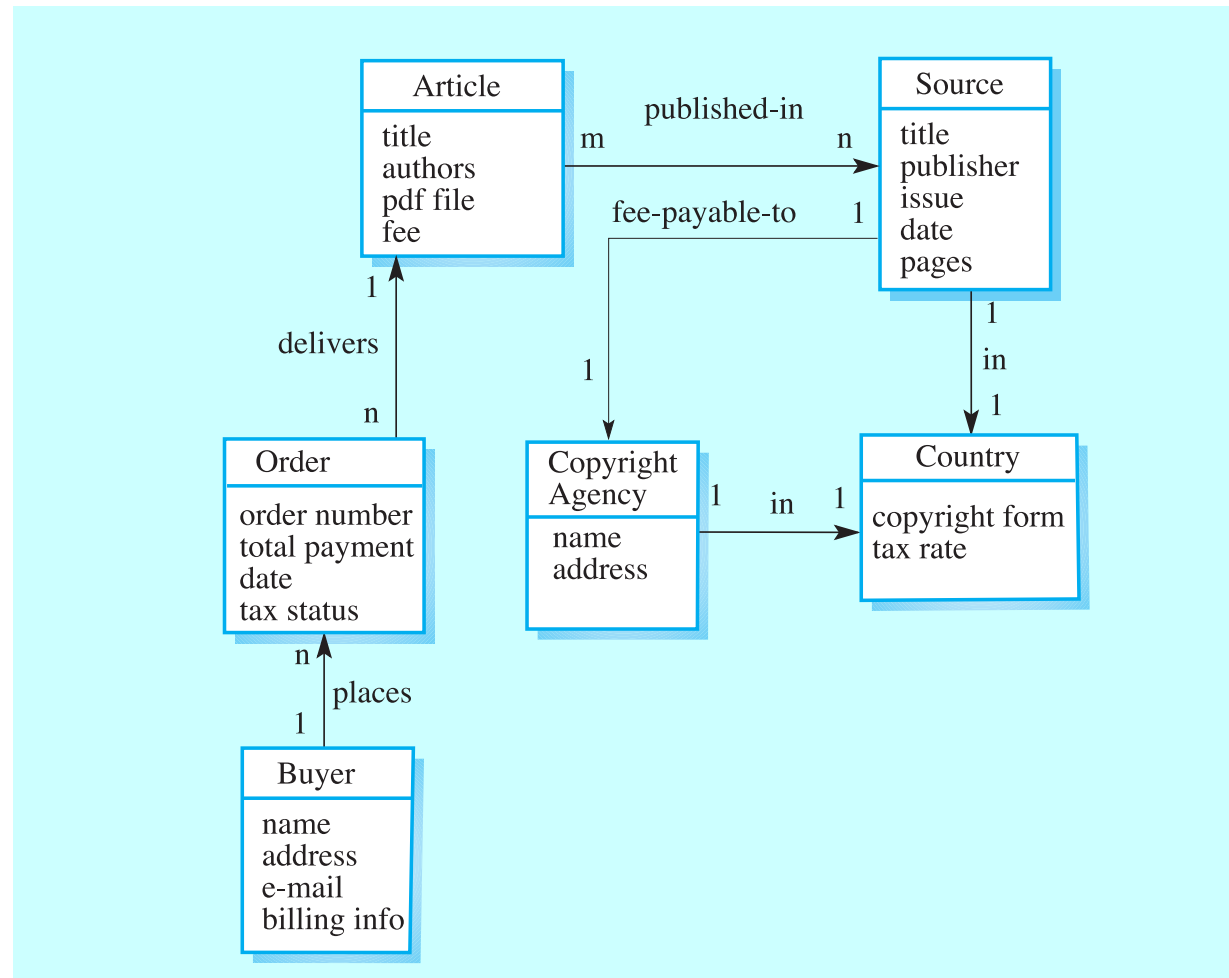
Figure 9-2: Data Flow Diagram (Yourdon Notation)



Durum-Makinesi (“State-Machine”) Modeli – Örnek: Mikrodalga Fırın Durum-Makinesi Modeli



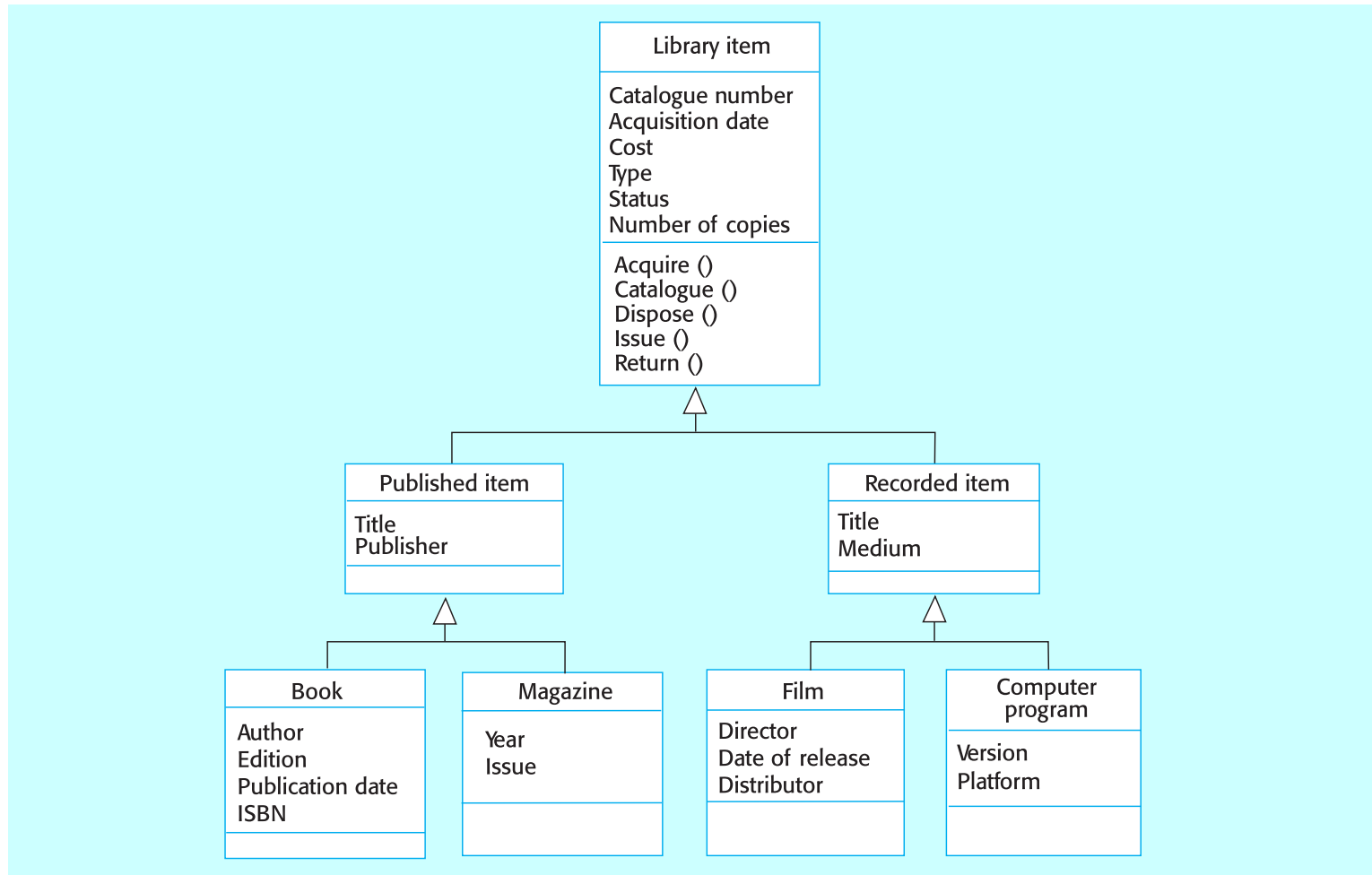
Veri (“Data”) Modeli – Örnek: Kütüphane Anlamsal Modeli



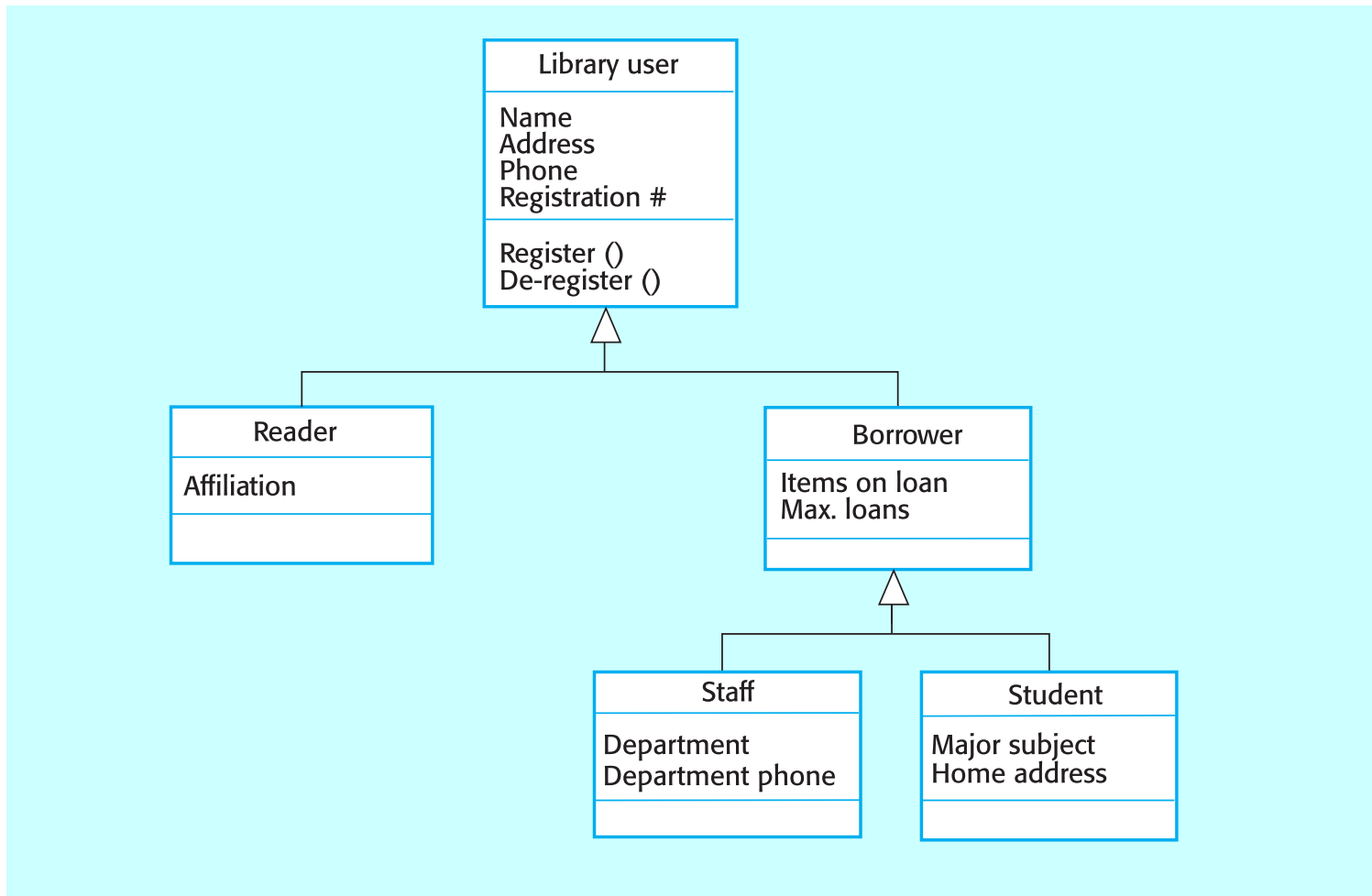
Veri (“Data”) Modeli – Örnek: Kütüphane Veri Sözlüğü (“Data Dictionary”)

Name	Description	Type	Date
Article	Details of the published article that may be ordered by people using LIBSYS.	Entity	30.12.2002
authors	The names of the authors of the article who may be due a share of the fee.	Attribute	30.12.2002
Buyer	The person or organisation that orders a copy of the article.	Entity	30.12.2002
fee-payable-to	A 1:1 relationship between Article and the Copyright Agency who should be paid the copyright fee.	Relation	29.12.2002
Address (Buyer)	The address of the buyer. This is used to any paper billing information that is required.	Attribute	31.12.2002

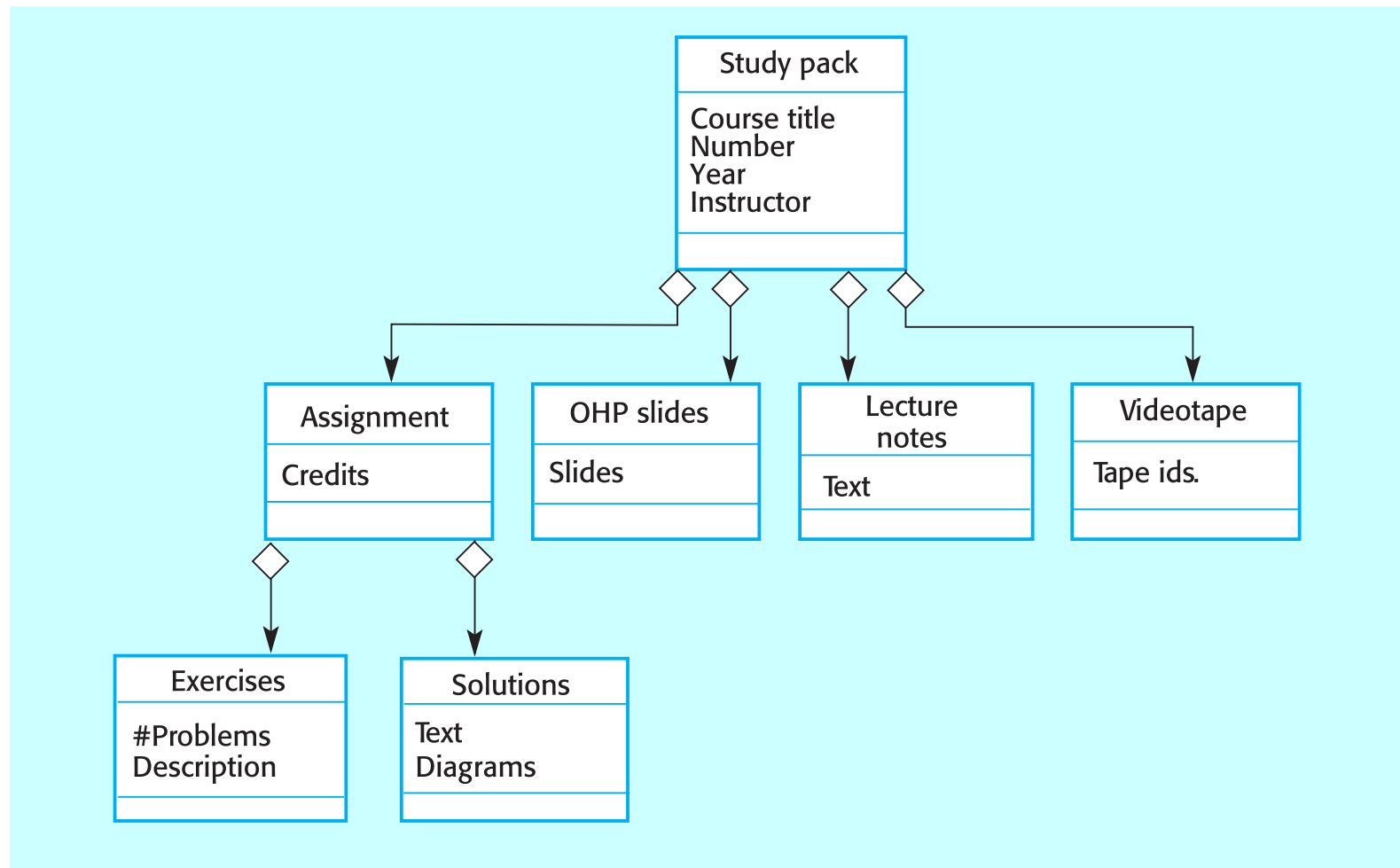
Nesne Kalıtım (“Object Inheritance”) Modeli – Örnek: Kütüphane Sınıf Hiyerarşisi



Nesne Kalıtım (“Object Inheritance”) Modeli – Örnek: Kütüphane Kullanıcısı Sınıf Hiyerarşisi Modeli



Nesne Bileşen (“Object Aggregation”) Modeli – Örnek: Ders Nesnesi Bileşen Modeli



Nesne Davranış (“Object Behavior”) Modeli – Örnek: Kütüphaneden Elektronik Öğe İndirme Davranış Modeli

