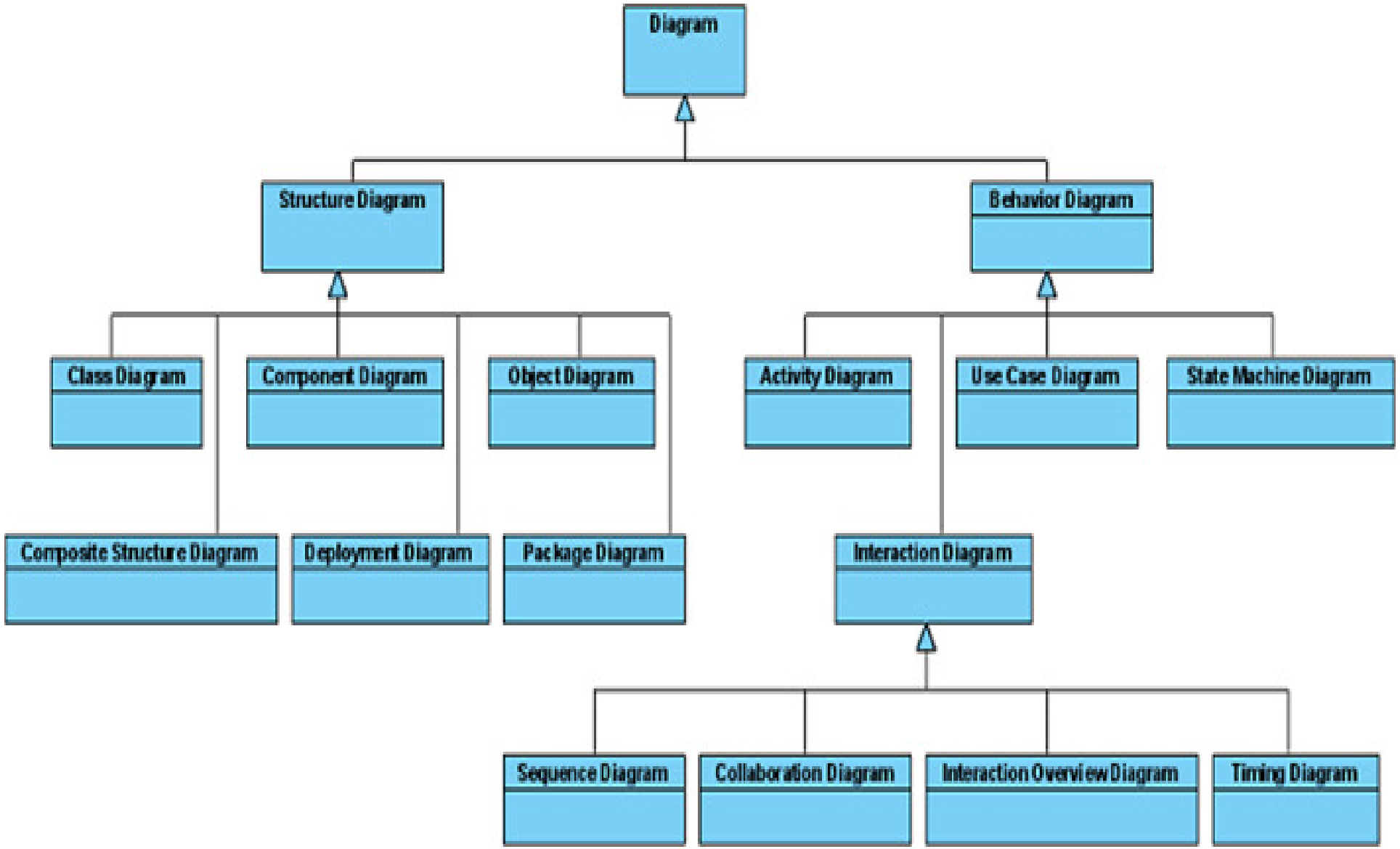
UML Diyagramlarının Genel Sınıflandırması



Davranışsal Diyagramlar

(Behavioral Diagramlar)

Use Case Diyagramı

Use Case Diyagramları

- Use case diyagramları «use case»ler , aktörler ve bunlar arasındaki ilişkileri simgeler. Böylece:
- Sistemin «use case» bakış açısından çözümü elde edilir.

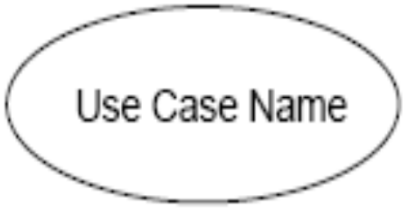
Herhangi bir «use case» incelenen sistemin herhangi bir fonksiyonelliğini betimler.

- «Use case» diyagramları farklı fonksiyoneller arasındaki ilişkileri ve bunların içsel/dışsal kontrollerini tanımlamak üzere kullanılır.
- Bu kontroller aktörler (actors) olarak adlandırılır. .

Use Case

- Use case, bir varlığın (entity) herhangi bir davranışını tanımlar.
  - Bu tanımlamada varlığın iç yapısı ile ilgili bir gösterim yoktur.
- Use case bir dizi eylemin (actions) betimlenmesidir.
  - Bu bir dizi eylem dışsal varlık (external entity) ile etkileşimde olabilir.
    - ✓Bir dizi eylem, senaryo eylemlerini ve hataların sıralanışını da içerebilir.

- Grafik gösterimi :



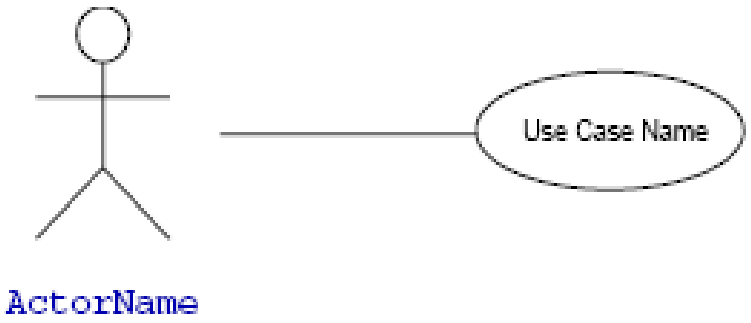
Aktörler

- Herhangi bir aktör «use case» ler ile etkileşim halinde iken, bu «use case» lerin birbirleri ile uyumlu rollerini tanımlar.
- Herhangi bir aktör, insan, donanım, ya da farklı bir sistem olmak üzere hem canlı hem de cansız bir varlıktır.

Aktörler ve «Use Case» ler arasındaki Associations (birliktelik) ilişkisi

Bir «use case» diyagramı aktörlerle bir dizi «use case» ve bunlar arasındaki ilişkileri gösterir.

- Bir aktörün örnekleri ile , bir «use case»in örneklerinin birbiri ile iletişim haline olması durumunda birliktelik (association) ilişkisi tanımlanmıştır.



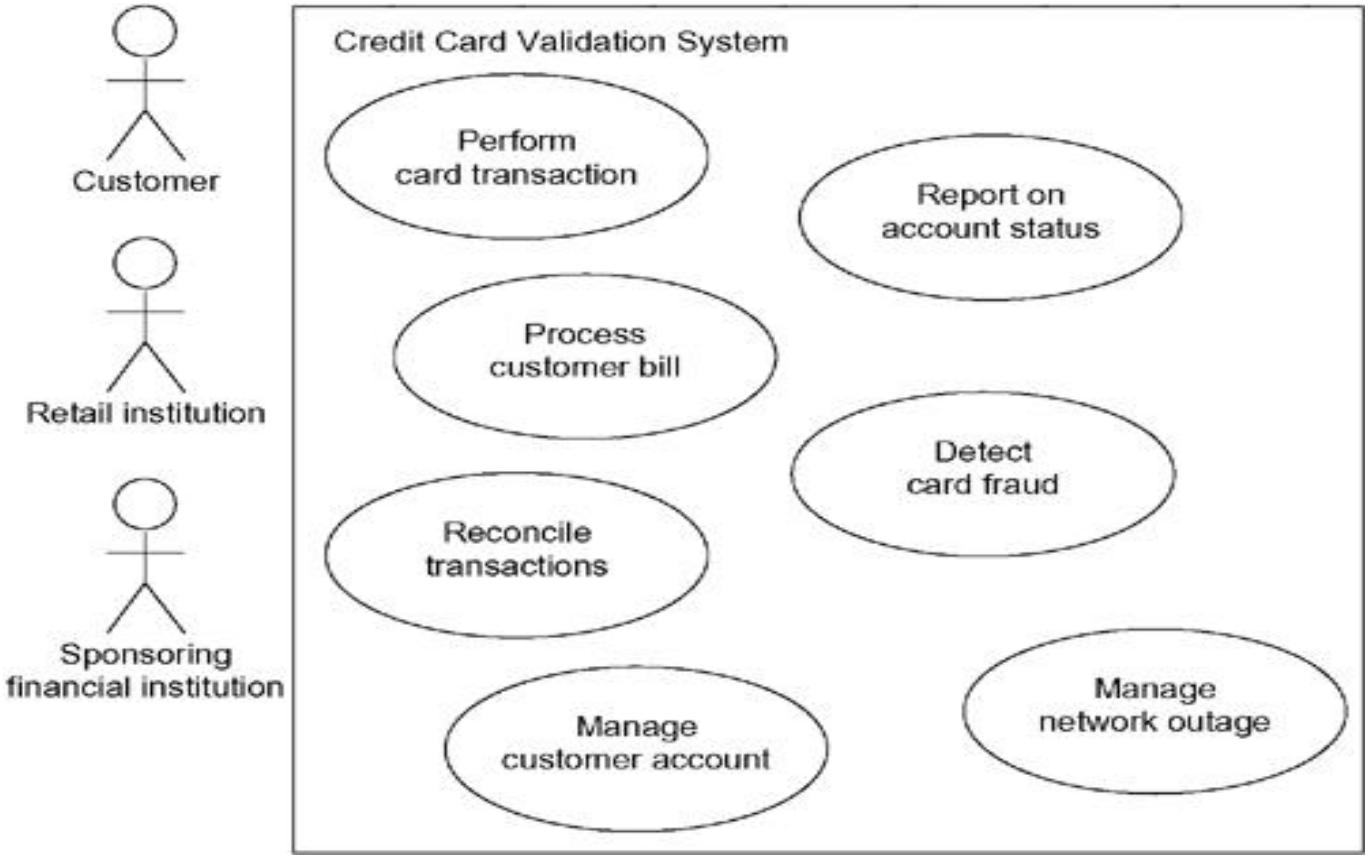
Aktörler ve «Use Case»ler arasındaki İletişimler

- Bir aktör bir varlığa ait pek çok «use case» ile iletişimde olabilir.
- Bir «use case» işlevini gerçekleştirirken bir veya daha fazla aktör ile iletişimde olabilir.
- İki «use case» aynı varlığı betimleyerek birbiri ile iletişimde bulunamaz.
  - Çünkü «use case» lerin herbiri varlığı tek başına kullanır.
    - ✓İki use case birbirleri ile iletişimde ise , include ya da extend ilişkisindeki use case aktörle bağıntılı olamaz. Aktörle ilişki diğerinden gelir.

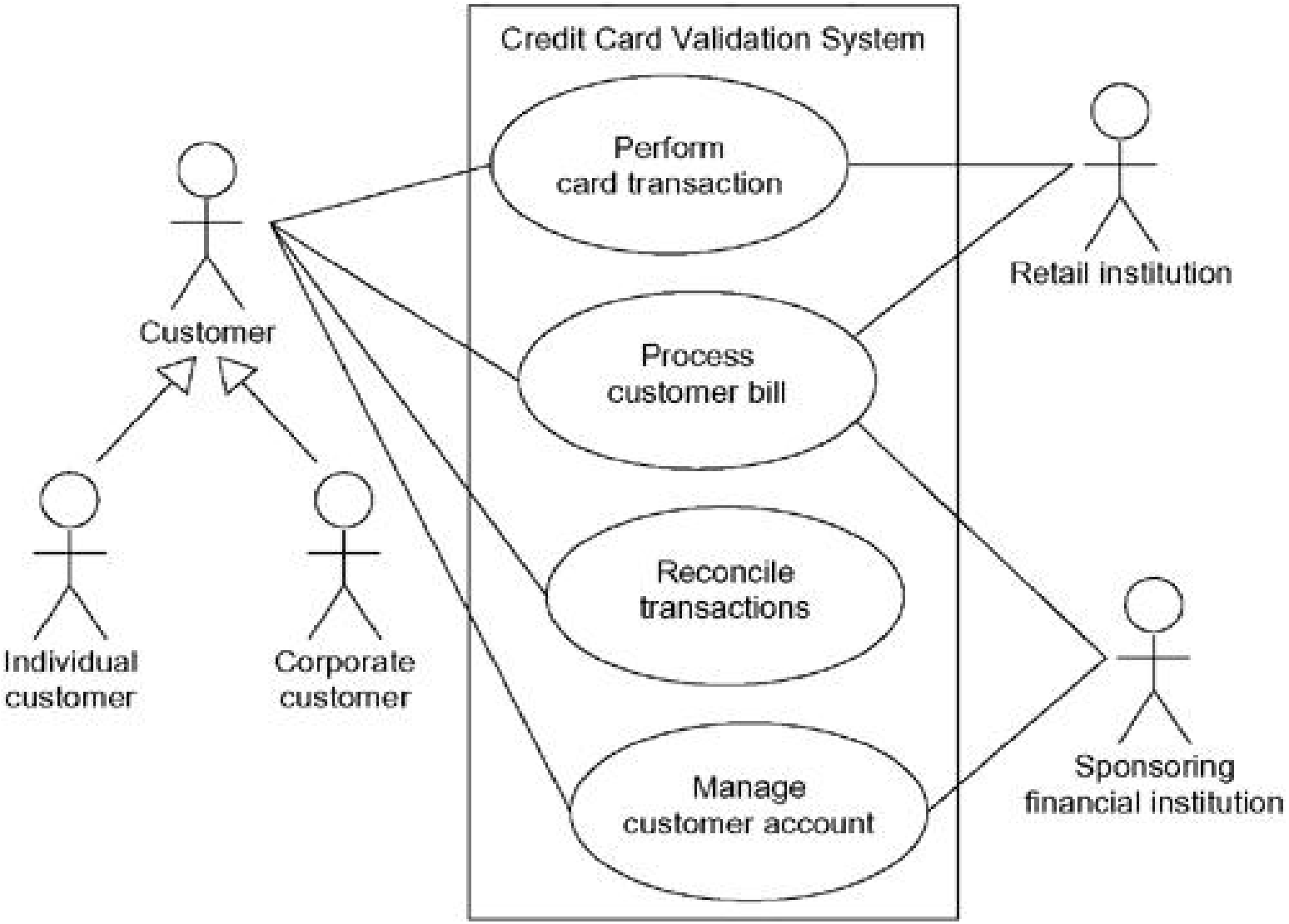
«Use Case»lerin Tanımlanması

- Bir «Use case» olası işlevlerini tanımlarken her bir aktörün aşağıdaki soruları cevaplaması beklenir.
  - Bunlar ne (what) sorularıdır.
    - ✓Aktör ne gerçekleştirmek istemektedir?
    - ✓Aktörün neler yapabilme kapasitesi vardır?
    - ✓Aktörün temel görevi (task) nedir?
    - ✓Aktörün sistemden alması gereken bilgiler nedir?
    - ✓Aktör sisteme ne bilgiler sağlar?

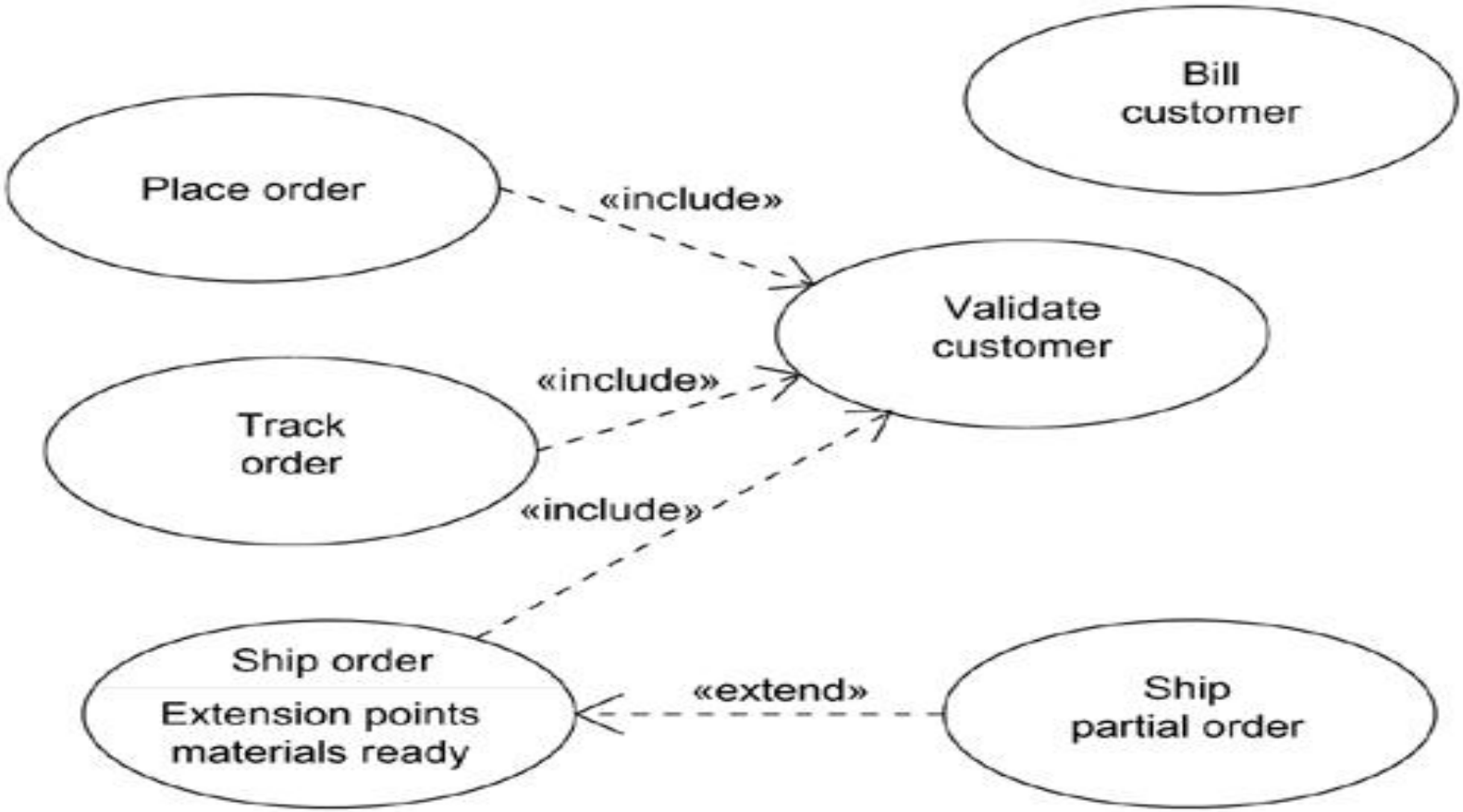
Bir Yazılım Sisteminin Gereksinimlerinin Modellenmesi



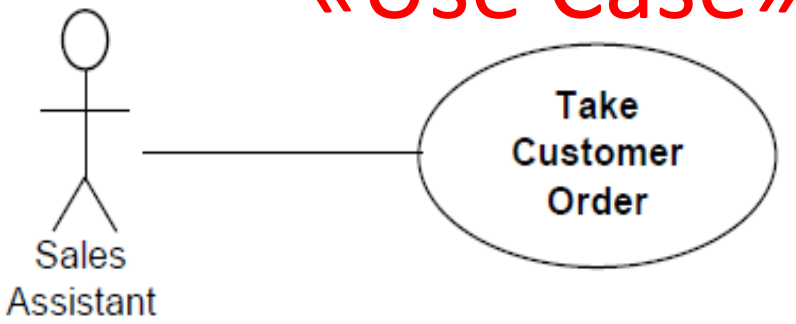
Sistem Bileşenlerinin Modellenmesi



Herhangi bir Elemanın Davranışının Modellenmesi



«Use Case» diyagramı örneği



Use Case: «Take the Customer Order» (Müşteri Siparişinin alınması

Temel Akış: 1. Aktör , müşteri ayrıntılarını girer

2. Aktör ürün için gerekli kodu girer.

3. Sistem gerekli miktarı görüntüler

4.Aktör gerekli miktarı girer.

5. Aktör ödeme ayrıntılarını girer.

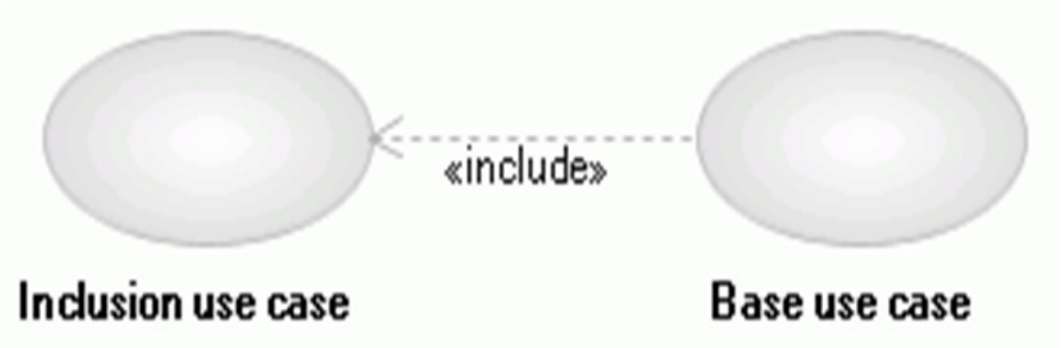
6. Sistem müşteri siparişini kaydeder.

Alternatif Akış: 4. adımdan sonra, aktör gerekli miktarı girdiğinde, Adım 2 ile Adım 4 arası ürün ilavesi için tekrarlanır.

5. Ödeme ayrıntılarını girmek için 5. adım tekrarlanır

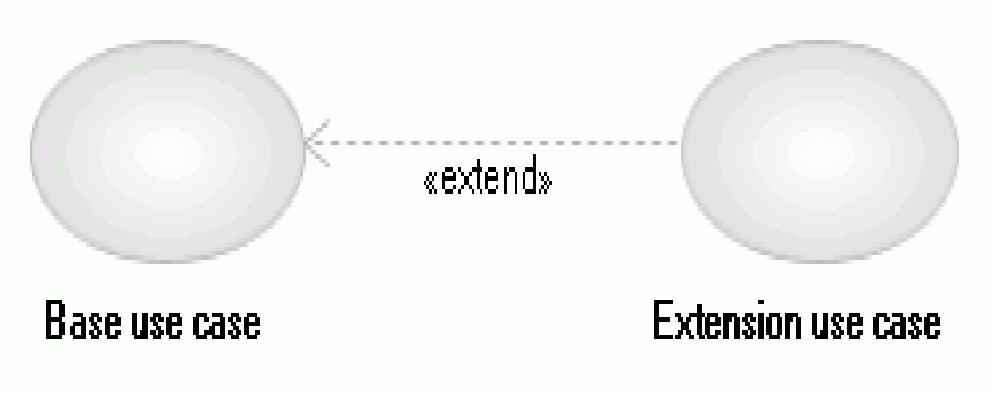
«use case»ler arasında ilişkiler

- «Use Case» ler arasında <<include>> ve <<extend>> ilişkileri gerçekleşebilir.

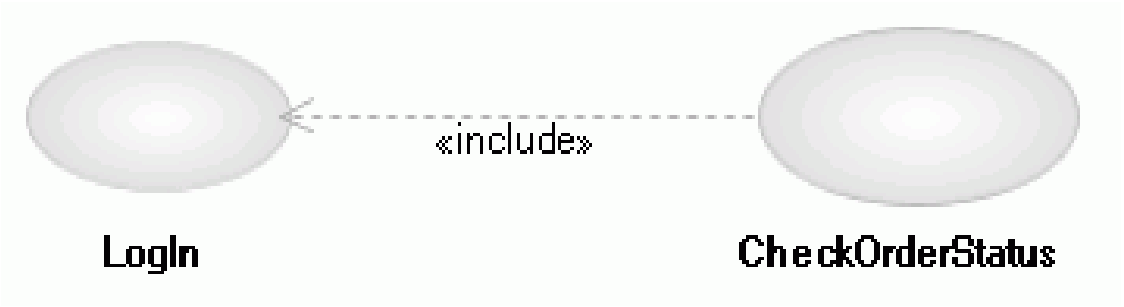


«Base use case» işlevini tamamlamak için mutlaka «inclusion use case» içindeki işlevleri gerçekleştirmelidir.

«Base use case» işlevini tamamlamak için «extension use case» içindeki işlevleri gerçekleştirmek zorunda değildir.



<<include>> ilişkisi örneği



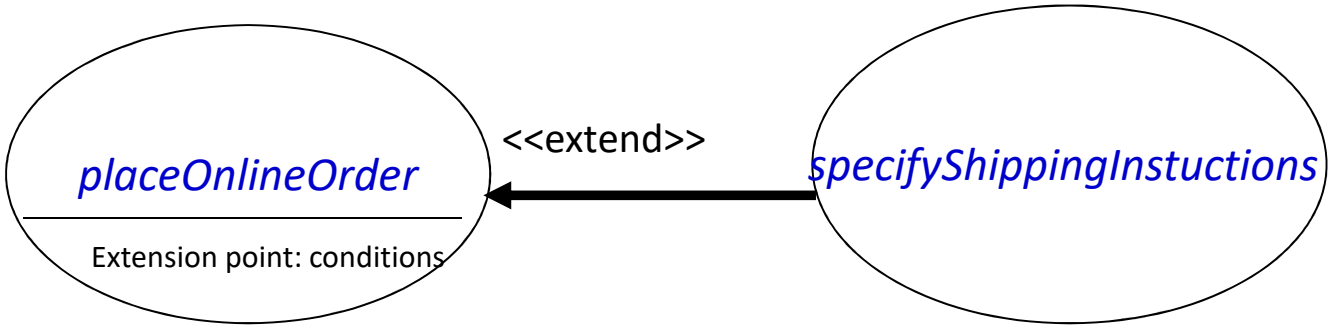
CheckorderStatus isimli use case, *Login* isimli use case in gerçekleştiği işlevleri içerecektir

include ilişkisi sadece «use case» ler arasında gerçekleşir.

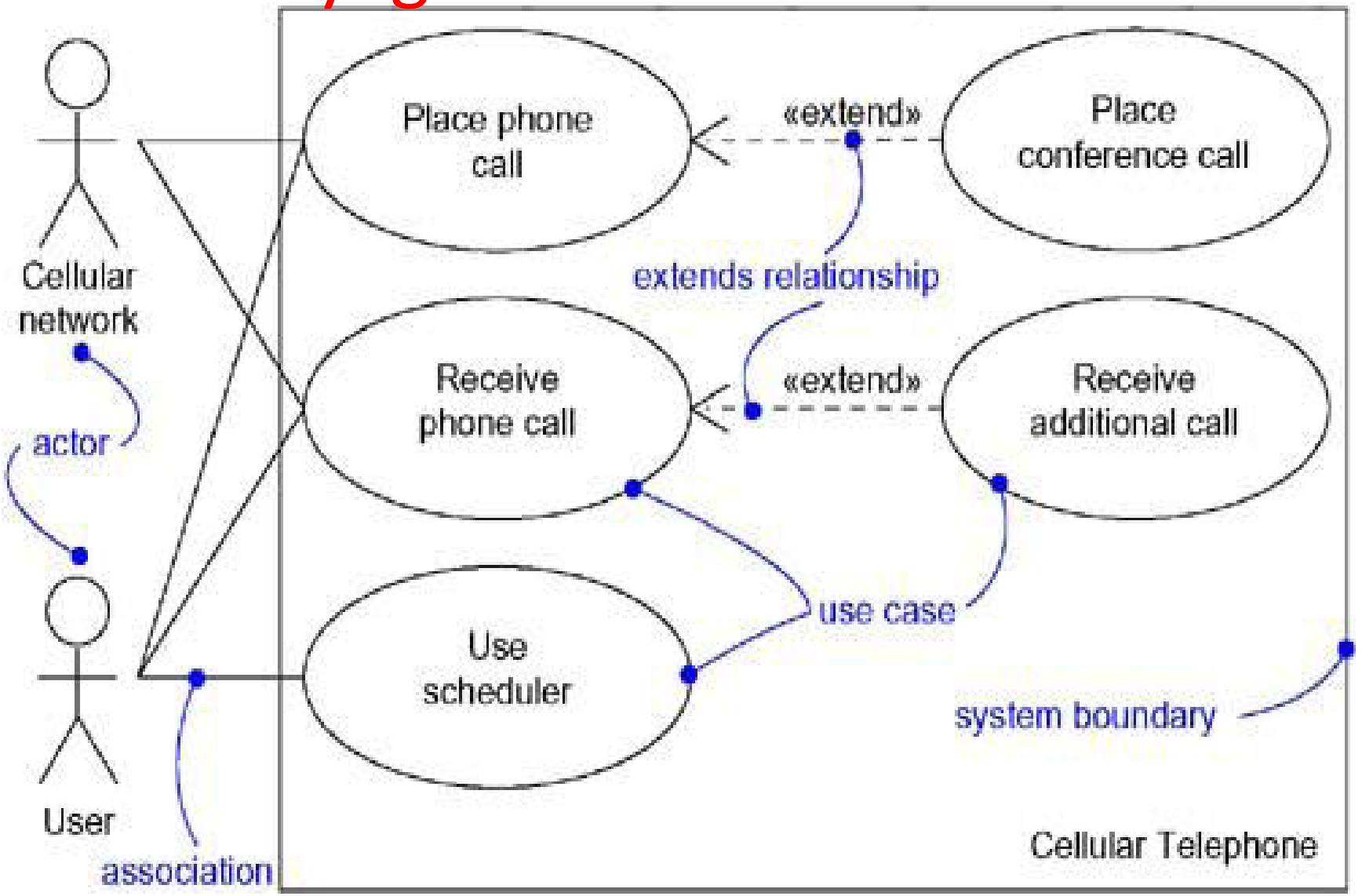
<<extend>> ilişkisi örneği

Base use case olan *placeOnlineOrder, specifyShippingInstuctions* isimli bir «use case»e extend ilişkisi ile bağlıdır.

specifyShippingInstuctions isimli *use case, placeOnlineOrder* isimli *«use case» içerisinde sadece koşullu olarak yer alacaktır; yani belli koşullar sağlandığında işlevli olacaktır.*



Örnek Diyagram

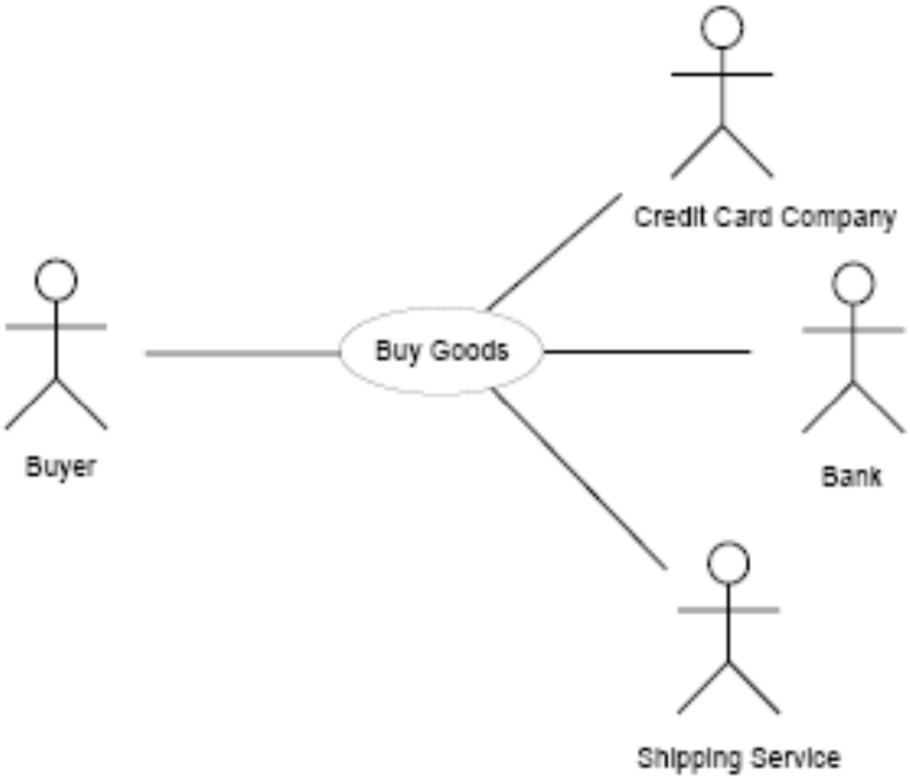


«include» ve «extend» ilişkileri arasındaki Temel Farklar

	İnclude use case	Extend use case
Bu «use case» seçmeli midir?	No	Yes
«Base use case» bu «use case» olmadan işlevini tamamlar mı?	No	Yes
Bu «use case» in çalışması koşula bağlı mıdır?	No	Yes
Bu «use case» «base use case» in davranışını değiştirir mi?	No	Yes

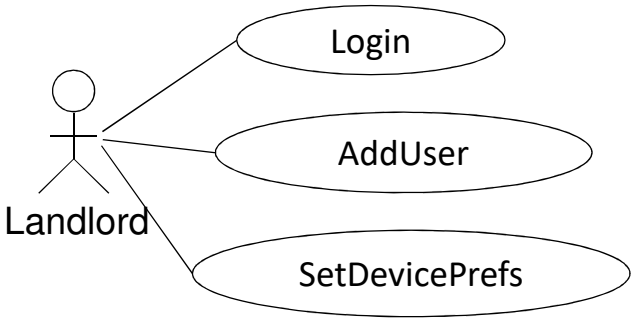
Use Case Diyagramı
«Online Bookshop» örneği

Use Case Diyagramı
«Buy Goods» Örneği

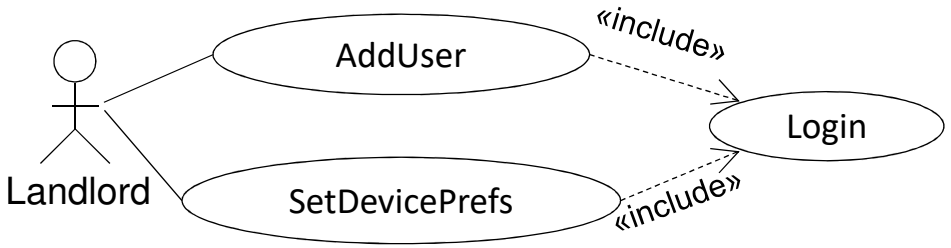


Use case Diyagramı
«Login» Örneği

BAD:



GOOD:



Davranışsal Diyagramlar II

(Behavioural Diagrams)

«Sequence» Diyagram

Sequence Diyagramı

- «Sequence» diyagramı bir nesneden (object) diğer nesneye aktarılan mesajları gösterir.
- Bu akışlar fonksiyon çağrıları şeklinde gerçekleştir.
- Böylece «sequence diyagramı», belli bir işlevi (fonksiyonelliği gerçekleştirmek üzere) sistem tarafından gerçekleştirilen bir dizi çağrıdan meydana gelir.

.

Nesneler, ömür çizgisi, mesajlar
(Objects, Lifelines, Messages)

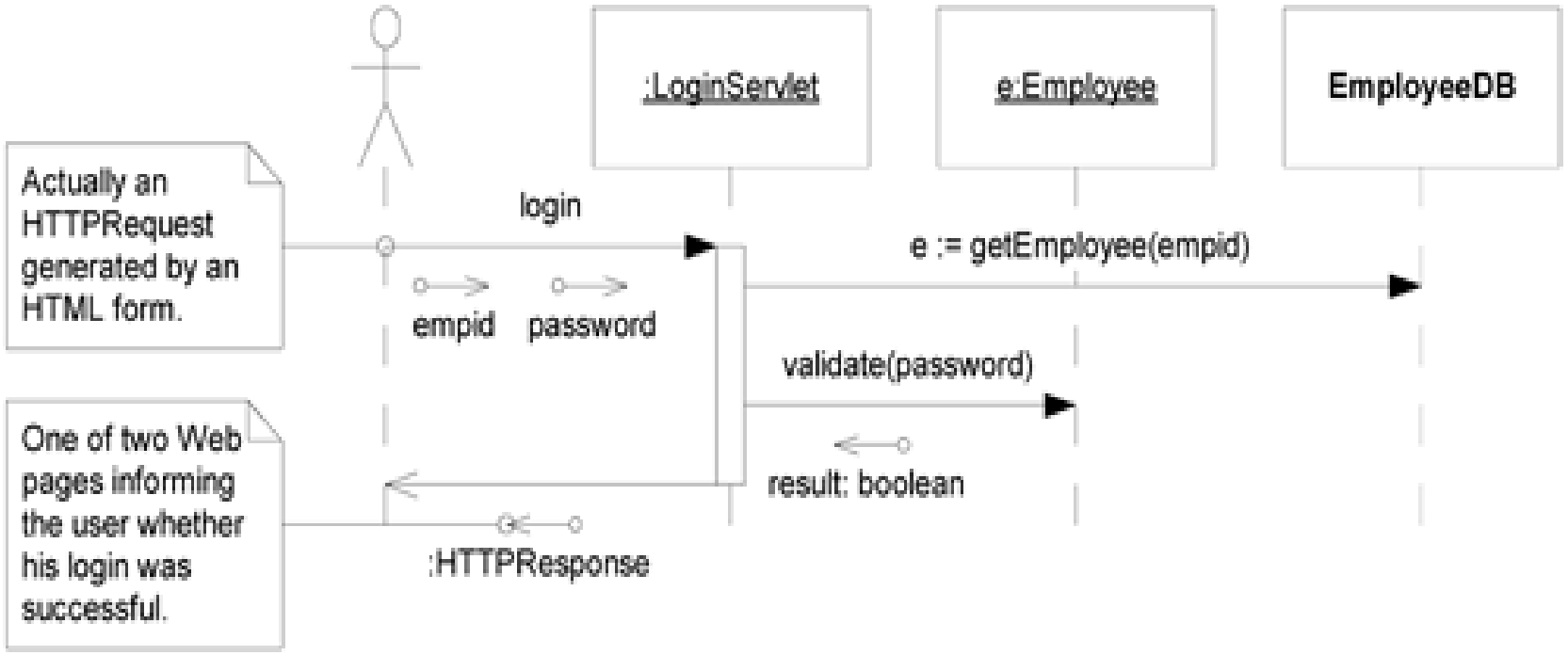
- Nesneler ve sınıflar en üstte birlikte bulunurlar.
- Nesneleri sınıflardan ayırmanın bir yolu nesnelerin altında çizgi olmasıdır.
- Çöp adam (aktör) solda isimsiz bir nesneyi simgeler.
  - Bu çöp adam kaynaktır (source) ve siteme giren ve çıkan tüm mesajları alır (biriktirir)
  - Tüm «sequence» diyagramlarında isimsiz aktör olmak zorunda değildir.

Nesneler, ömür çizgisi, mesajlar

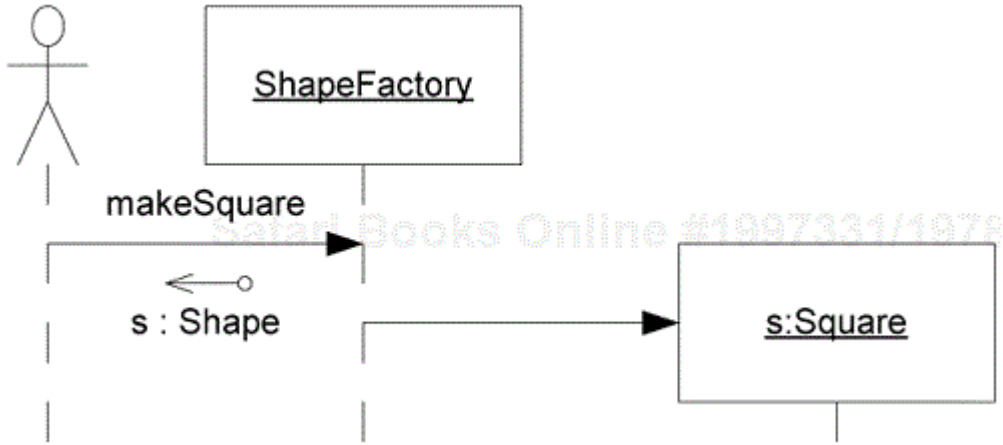
(Objects, Lifelines, Messages)

- Nesnelerden aşağıya kesikli çizgiler (dashed lines) aktörlerin ömrü olarak adlandırılır (lifelines).
- Bir nesneden diğerine gönderilen mesaj ok ile gösterilir (iki ömür çizgisı (lifeline) arasındaki oktur)
- Her mesajın bir ismi vardır (isim ile etiketlenmiştir)
- Argümanlar parantez içerisinde görüntülenirler.
- Zaman (Time) dikey boyuttur; böylece daha düşük bir mesaj gönderildikten sonra görünür.

Örnek 1
Sequence Diyagramı

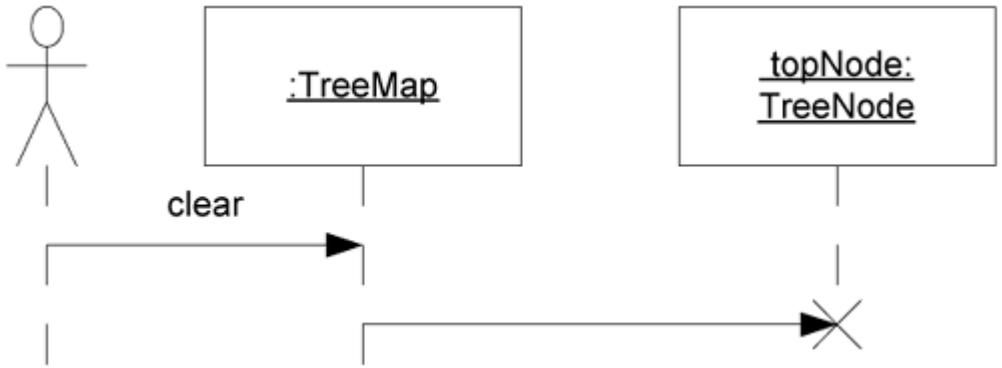


Örnek 2
Sequence Diyagramı



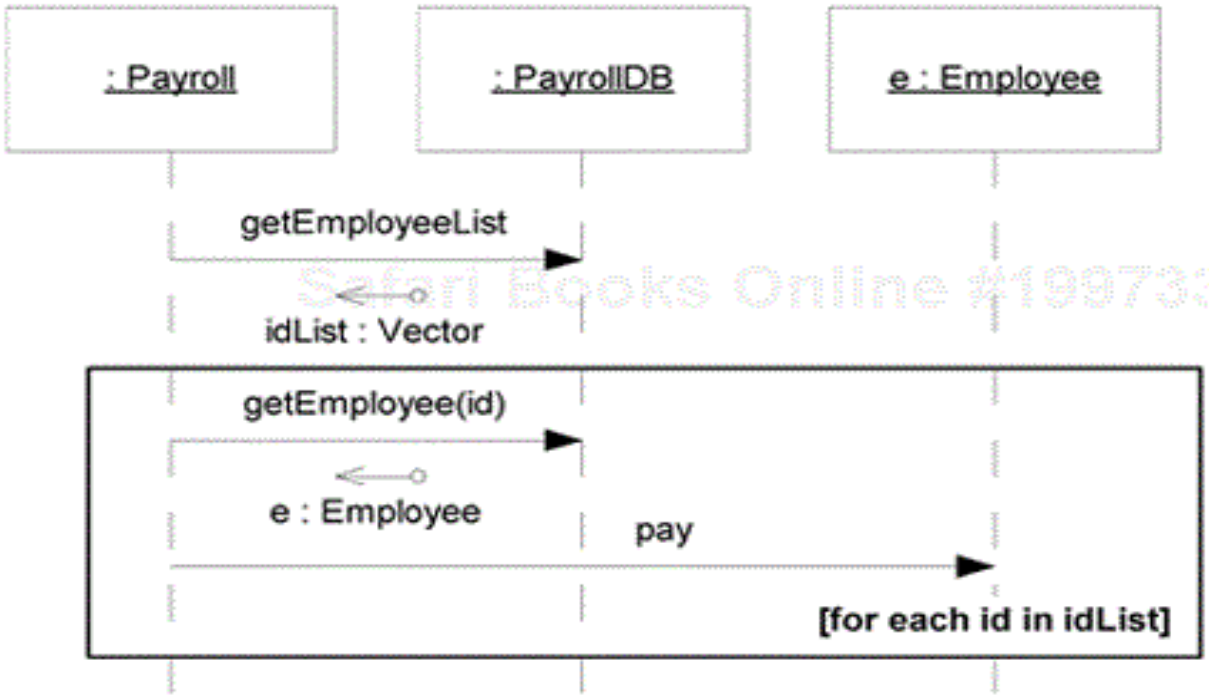
Bu diyagram bir nesne oluşturur (Creating an object).

Örnek 3
Sequence Diyagramı

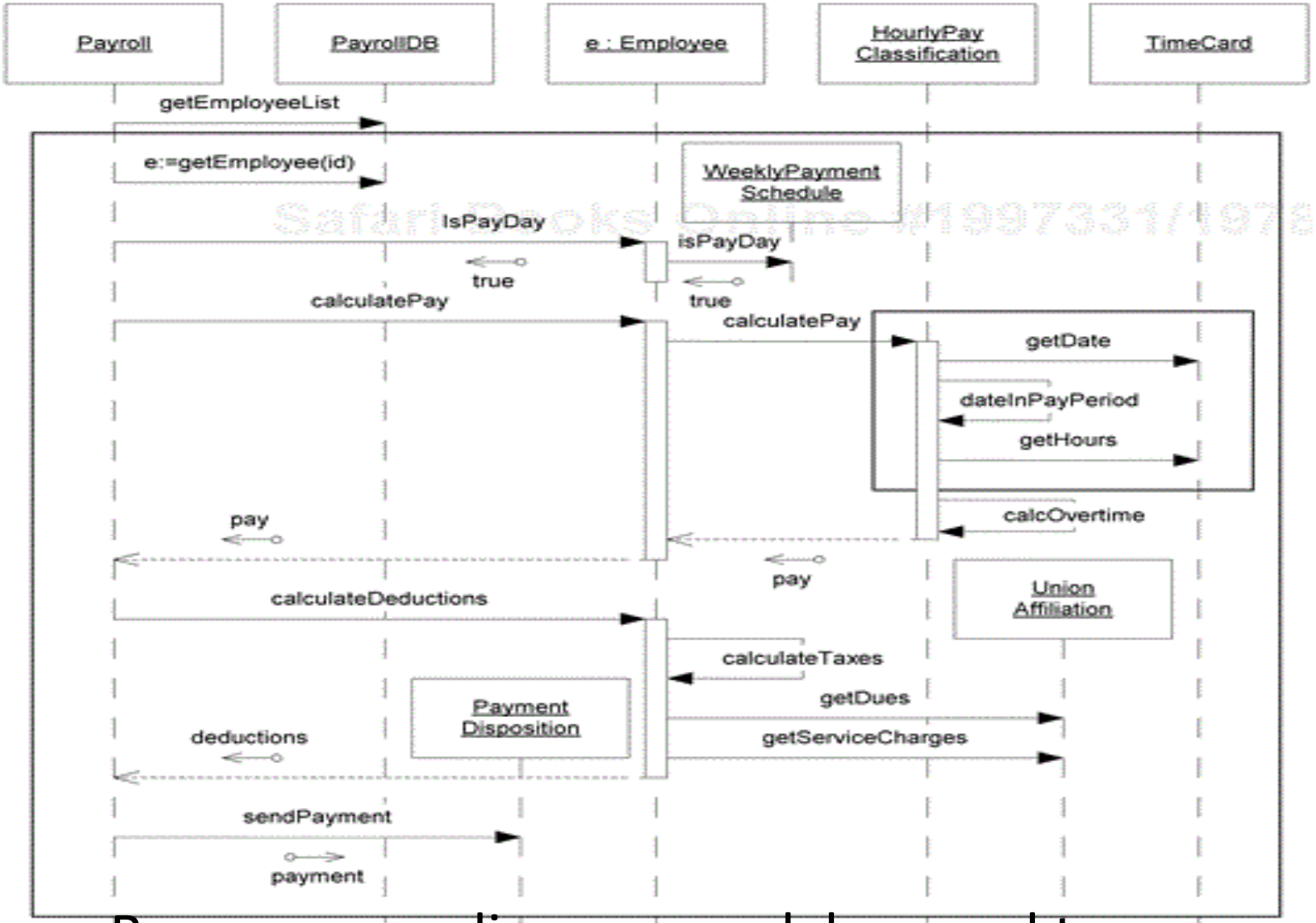


Bu diyagram bir nesneyi çöp kutusuna gönderir.
(Releasing an object to the garbage collector)

Örnek 4
Basit Döngüler (Simple Loops)



Örnek 4 ile ilgili
Farklı Durumlar ve Senaryolar

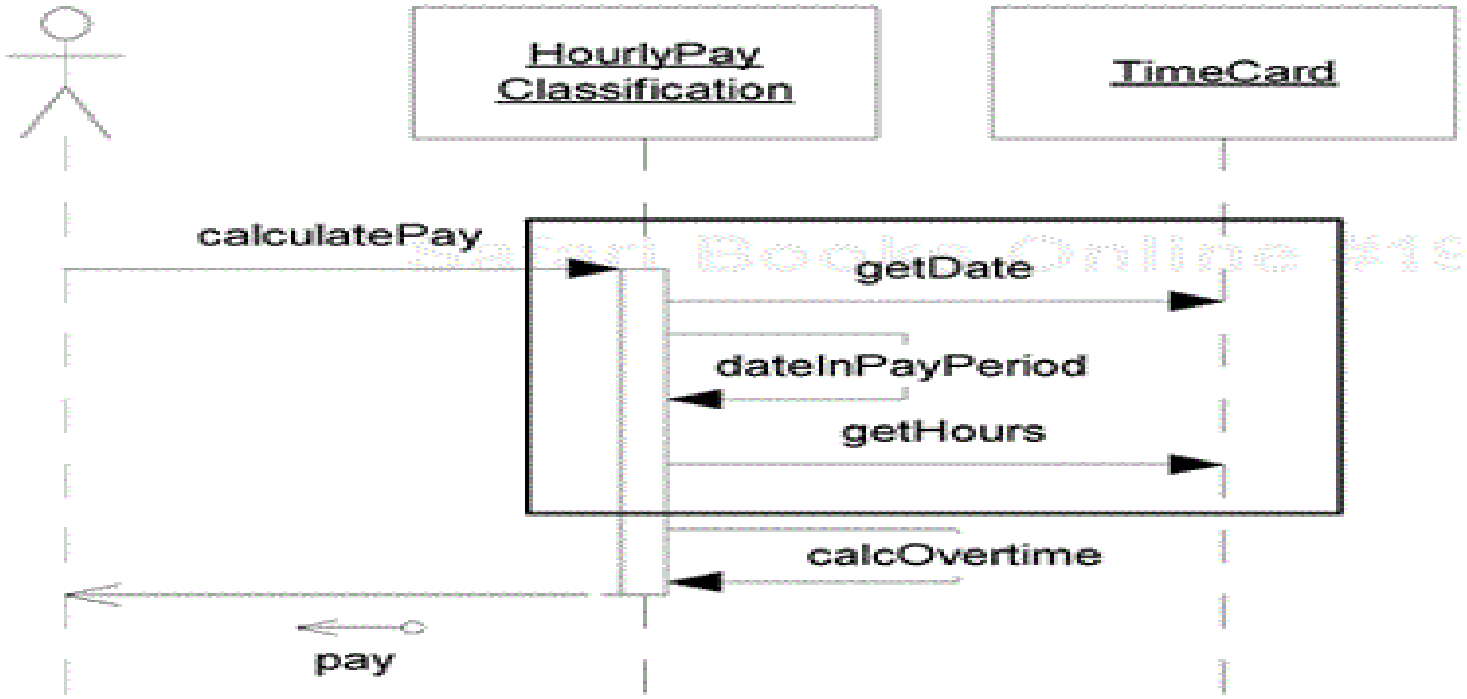


Bu sequence diyagramı çok karmaşıktır.

«Sequence» Diyagramı
Tasarlama Kuralları

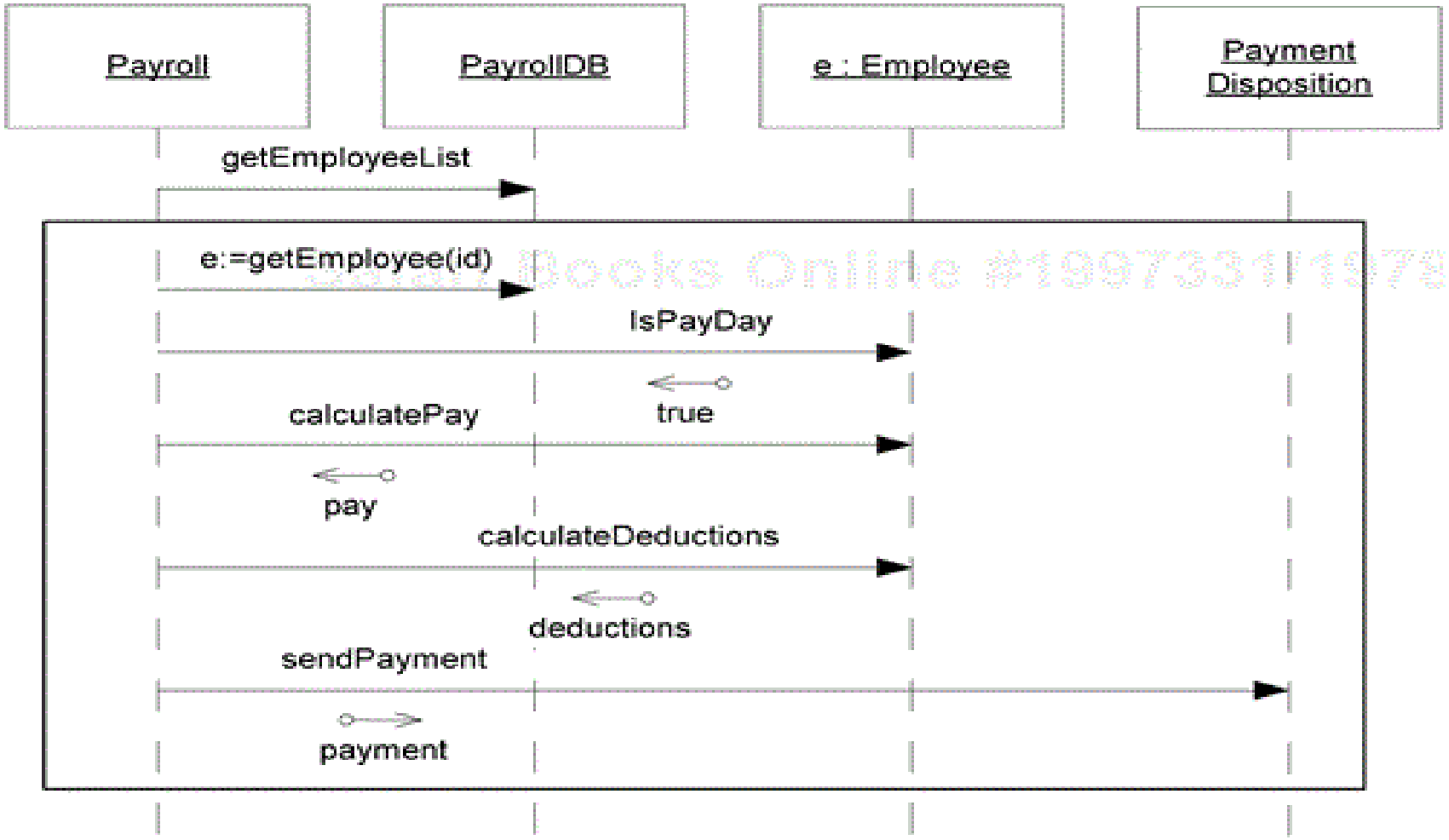
- sequence diyagramları kesinlikle karmaşık tasarlanmamalıdır
- Karmaşık bir diyagramı kimse okuyamaz .
- Karmaşık bir diyagramı okuyabilecek kişi olsa bile karmaşıklığından dolayı okumayacaktır.
- Bu da gereksiz bir zaman kaybıdır.
- O nedenle daha küçük «sequence» diyagramının nasıl tasarlanacağı öğrenilmelidir.
- Her «sequence» diyagramı tek bir sayfada tasarlanmış olmalıdır

Örnek 4 ile ilgili Farklı Durum



Sequence Diyagramı ile ilgili bir senaryo

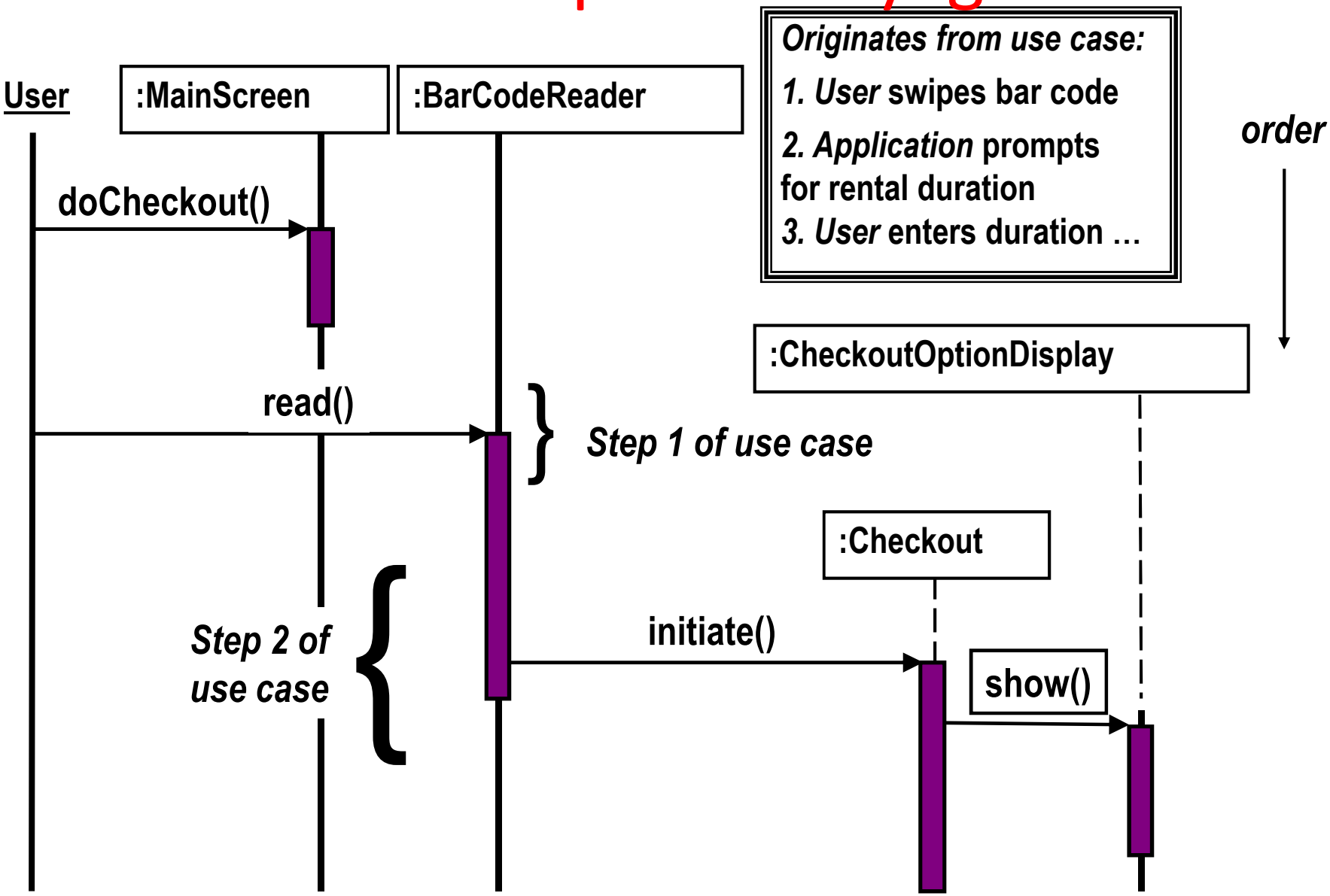
Örnek 4 ile ilgili
Farklı Durum ve Senaryo



Örnek 5
Sequence Diyagram

- ***Use case:***
- ***1. User* swipes bar code**
- ***2. Application* prompts for rental duration**
- ***3. User* enters duration**
- ***4. Application* stores record**
- ***5. Application* prints customer's account status**

Örnek 5 Sequence Diyagram



Örnek 5 Sequence Diyagram