HAFTA 4: GEREKSİNİM ANALİZİ

İŞ GEREKSİNİMLERİ ANALİZİ – GEREKSİNİM ANALİZİ İLİŞKİSİ

SİSTEMİN BAĞLAMI (CONTEXT)

- Sistemin bağlamı, sistemin içerisinde çalışacağı ortamı ve bu ortamın gereksinimlerini (arayüzler, birlikte-çalışabilirlik, kısıtlar, vb.) ifade eder.
  - Gereksinim tanımlamaları için sınırları ortaya koyar.

- Bağlam ("context") diyagramı, sistemin kendi ortamındaki konumunu ve diğer sistem ve süreçlerle olan etkileşimini tanımlamaya yarar.
  - Bir sistemin gereksinimlerini tanımlamaya başlamadan önce ilk oluşturacağımız, temel modellerden biridir.
  - Yazılım içeren sistemler için işin bağlamını anlamak, yazılım ürününün doğasından kaynaklanan zorluklar (soyutluk, değişebilirlik, insan duyarlılık, vb.) sebebiyle özellikle önemlidir.

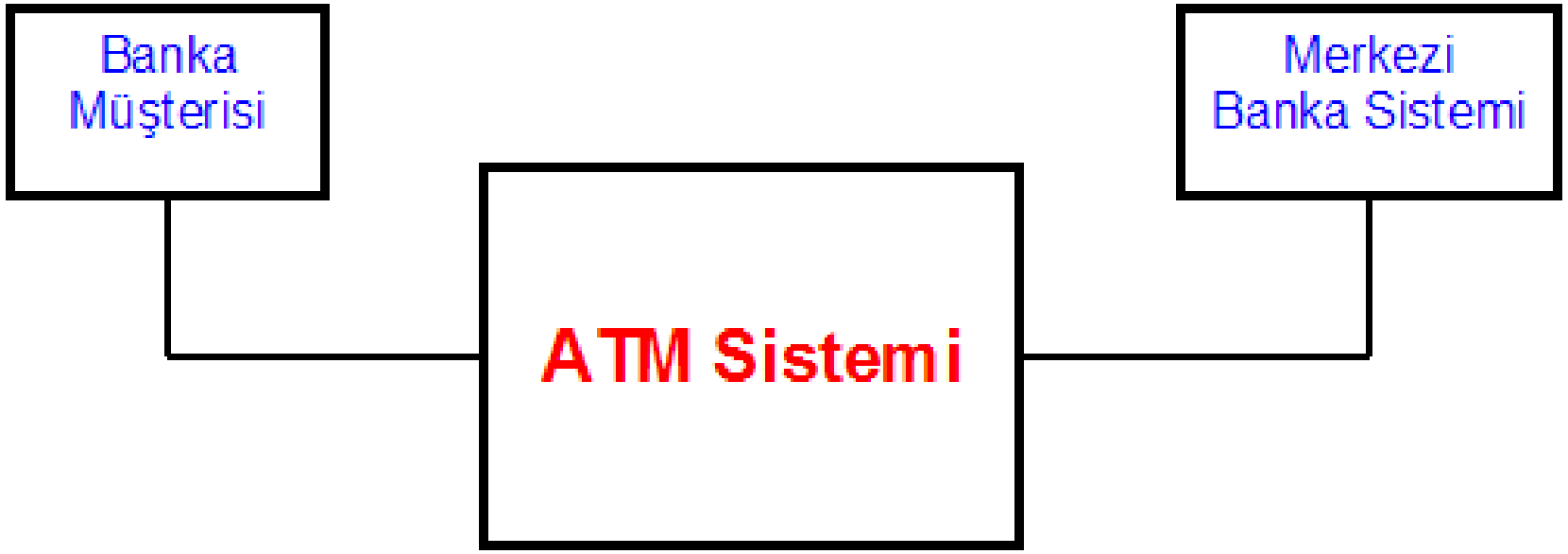
İŞ GEREKSİNİMLERİ ANALİZİ

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

ÖRNEK: ATM UYGULAMASI

- 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 UYGULAMASI – KAPSAM



Bağlam (“context”) diyagramı

İŞ GEREKSİNİMLERİ ANALİZİ

- 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 SÜRECİ (“BUSİNESS 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”

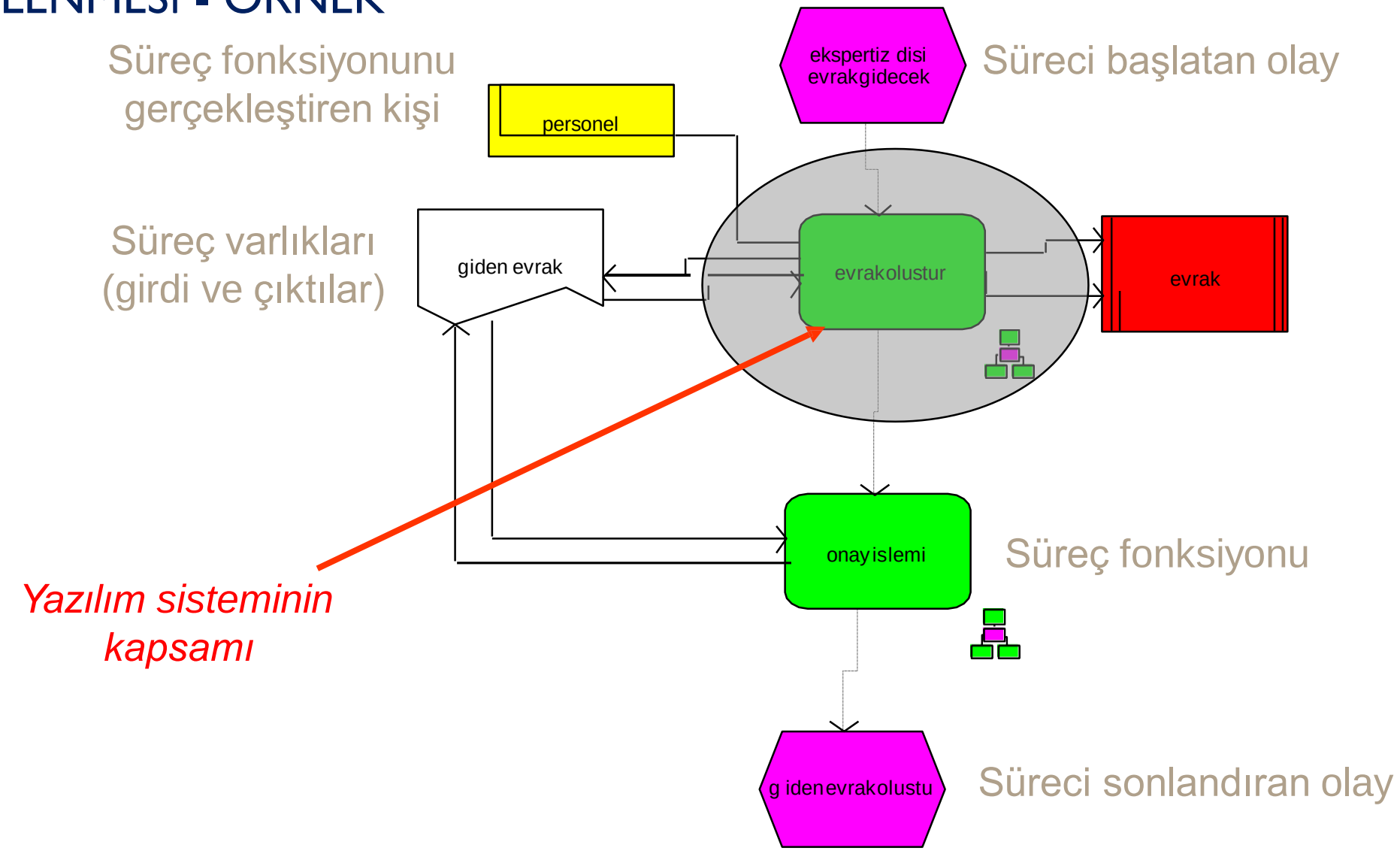
İŞ SÜREÇLERİNİN MODELLENMESİ

- 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 kullanıcılı olduğu durumlarda, getirileri maliyetini daha çok karşılar [Yourdon 2000]

İŞ SÜREÇLERİNİN MODELLENMESİ - ÖRNEK



İŞ SÜREÇLERİNİN MODELLENMESİ – 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 SÜREÇLERİ VE GEREKSİNİM ANALİZİ (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:
  - Geliştirilecek yazılım ürününden beklenen özellikleri *anlamak* ve *tanımlamak*
  - İş süreçlerinin tanımladığı bağlamda detaylı yazılım özelliklerini belirlemek

İŞ SÜREÇLERİ VE GEREKSİNİM ANALİZİ (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

GEREKSİNİM ANALİZİ – 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)

“USE-CASE” (KULLANIM VAKASI) ESASLI GEREKSİNİM ANALİZİ

İÇERİK

- Birleşik Modelleme Dili (“Unified Modeling Language - UML”)
- Gereksinim Analizi İçin Kullanılan Başlıca UML Elemanları
- “Use-Case” Esaslı Gereksinim Analizi
- Sınıf Çalışması: Kütüphane Destek Sistemi (KDS)

"UNIFIED MODELING LANGUAGE" - I

- Yazılım sistemlerinin modellemesi için geliştirilmiş standart bir dildir
  - Yazılım iş ürünlerinin; tanımlanması, görsel hale getirilmesi, belgelendirilmesi
  - Açık standarttır; birçok araç tarafından desteklenir
  - Tüm yazılım geliştirme sürecini destekler
- Çıkış hedefleri:
  - Kullanımı kolay, görsel bir modelleme dili sunmak
  - Programlama dillerinden ve geliştirme sürecinden bağımsız olmak
  - En iyi yöntemleri bütünleştirmek
- 1995 yılından beri sektörün katkıları ile gelişmektedir
  - Son sürüm: UML 2.5 (Haziran 2015), Object Management Group (OMG)
  - İlgili web sitesi: www.uml.org

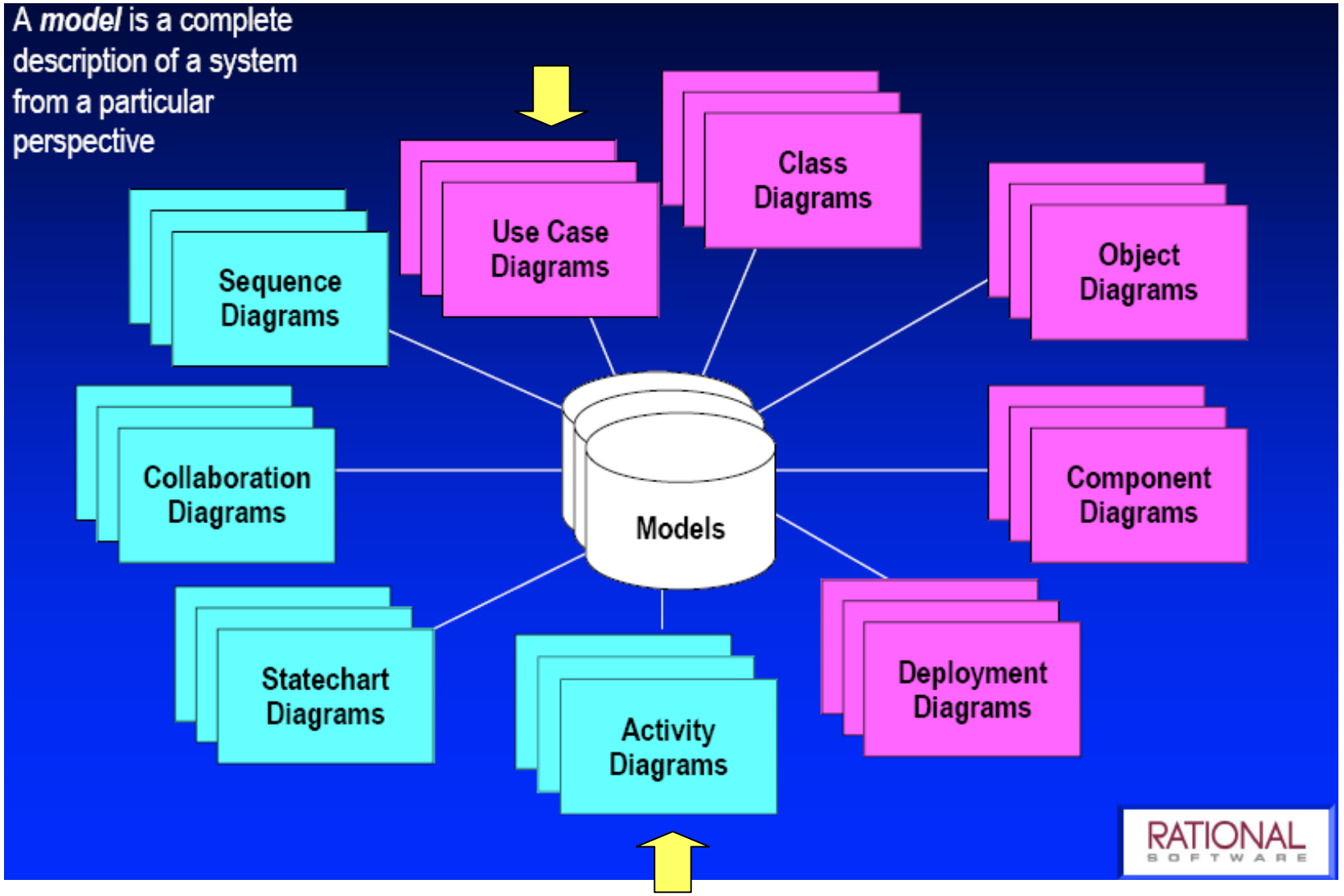
"UNIFIED MODELING LANGUAGE" - 2

- Sundukları:
  - Yazılım ürünlerinin gösterimi için yapı taşları ve ilişkiler
  - Yapı taşlarını ve ilişkileri kullanarak farklı bakış açılarını destekleyen diyagramlar
    - Tanımlanan yapı taşlarının ve diyagramların bütünü, geliştirilen yazılım sistemine karşılık gelir

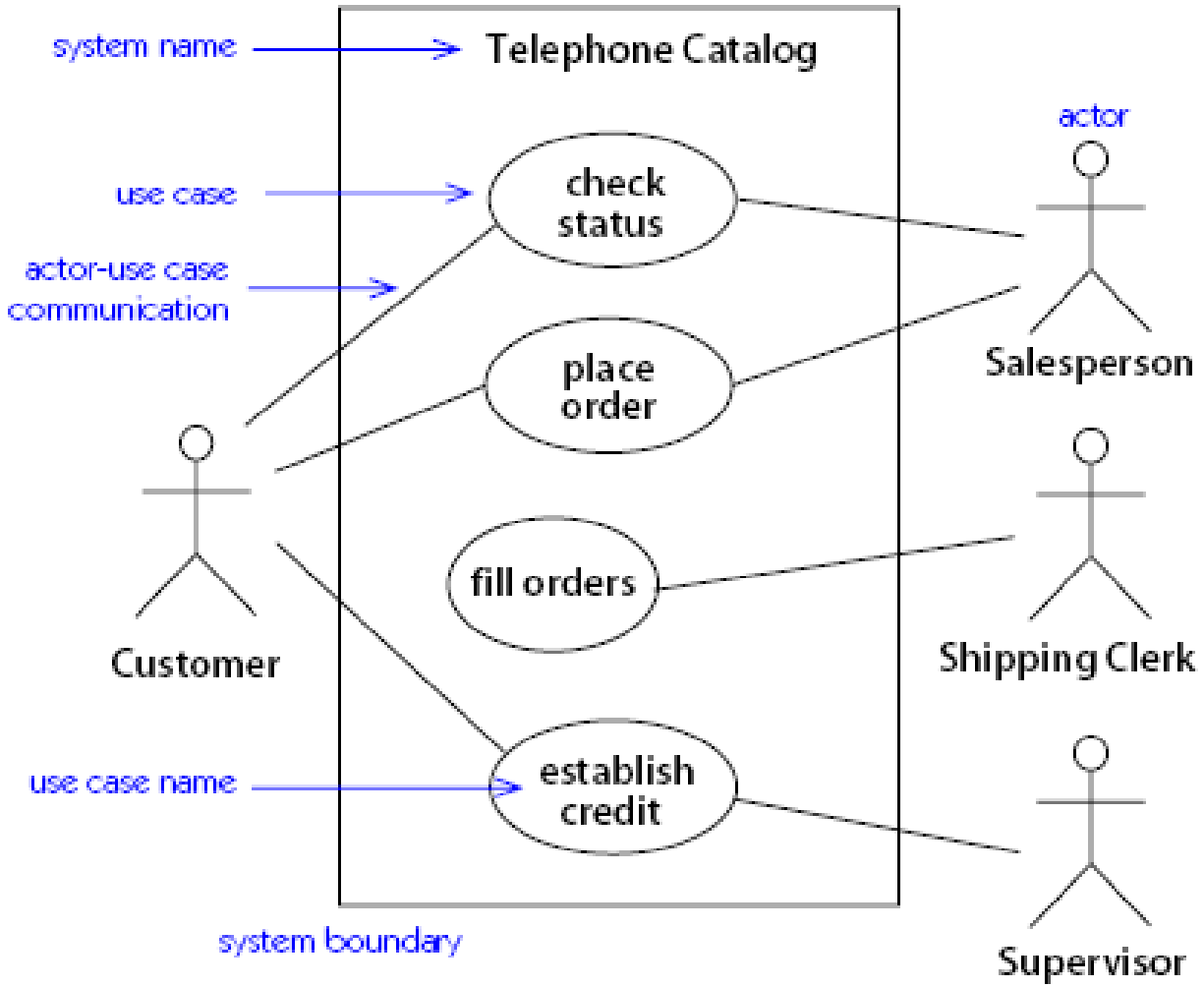
- Sunmadıkları:
  - Sisteminin nasıl geliştirilmesi gerektiğini tanımlamaz
  - Nesneye yönelik yazılım modellemesi için yapılar sunar, ancak;
    - Bu yapıların hangi sıra ile kullanılması gerektiğini tanımlamaz
    - Yapıların geliştirme sürecinin hangi aşamalarında kullanılması gerektiğini tanımlamaz

GEREKSİNİM ANALİZİ İÇİN KULLANILAN BAŞLICA UML ELEMANLARI

UML DİYAGRAMLARI

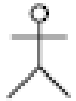


"USE-CASE DIAGRAM": ÖRNEK



"USE-CASE DIAGRAM" MODELLEME ÖĞELERİ

- Aktör



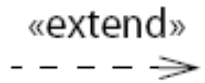
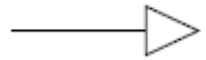
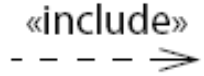
  - Sistemin kullanıcıları
  - "An outside user of a system"

- "Use-case"

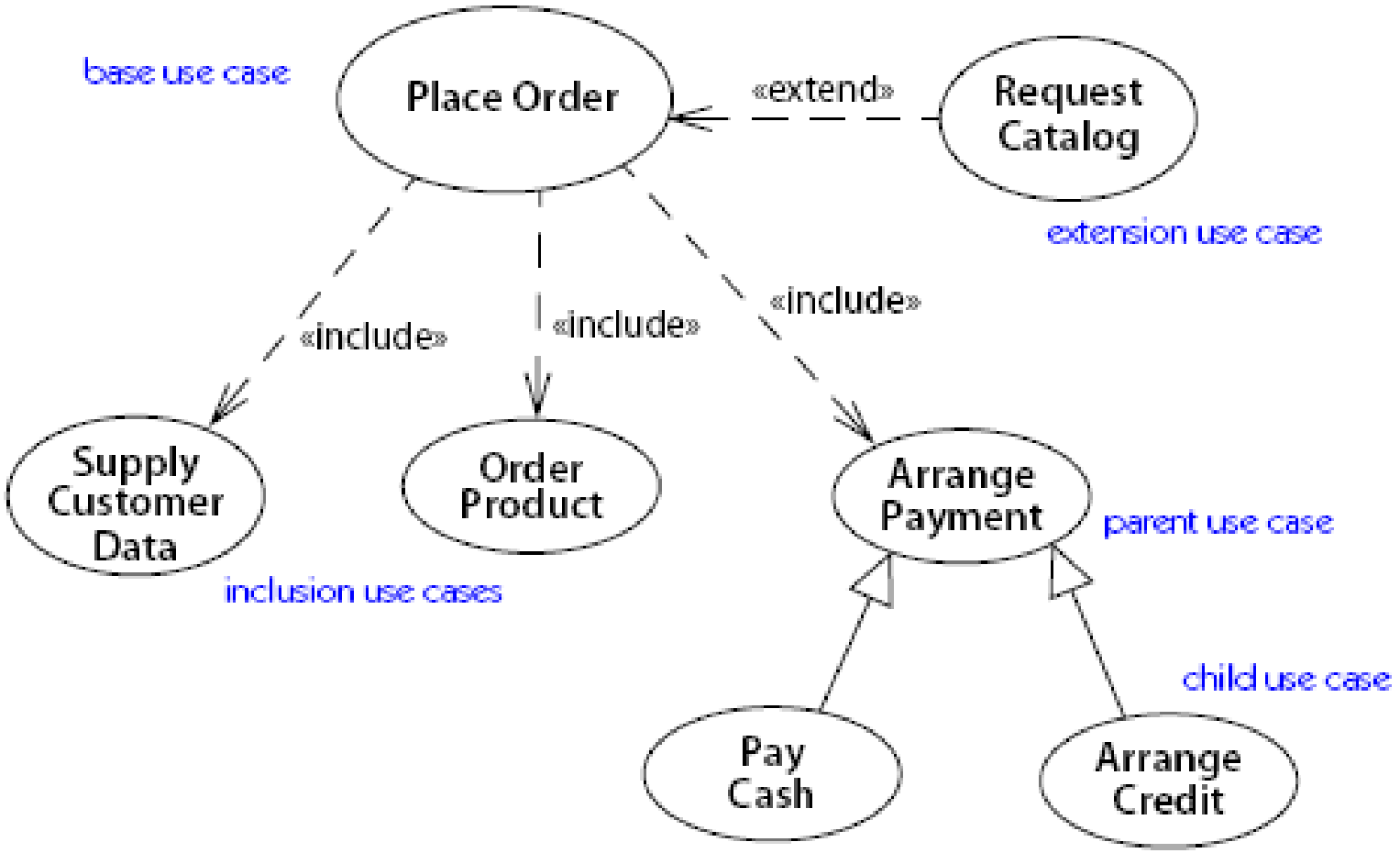
  - Sistemin destekleyeceği işler
  - A specification of the behavior of an entity in its interaction with outside agents

- İlişki ("relationship")
  - "Association": Aktör ve "use-case" arasındaki bağlantı
  - "Generalization": İki "use-case" veya iki aktör arasındaki kalıtım ilişkisi
  - "Extend": Bir "use-case"den diğerine geçiş (kontrol dışı)
  - "Include": Bir "use-case"in diğerinin davranışını içermesi

"USE-CASE" İLİŞKİ TÜRLERİ

Relationship	*Function*	*Notation*
association	The communication path between an actor and a use case that it participates in	
extend	The insertion of additional behavior into a base use case that does not know about it	
use case generalization	A relationship between a general use case and a more specific use case that inherits and adds features to it	
include	The insertion of additional behavior into a base use case that explicitly describes the insertion	

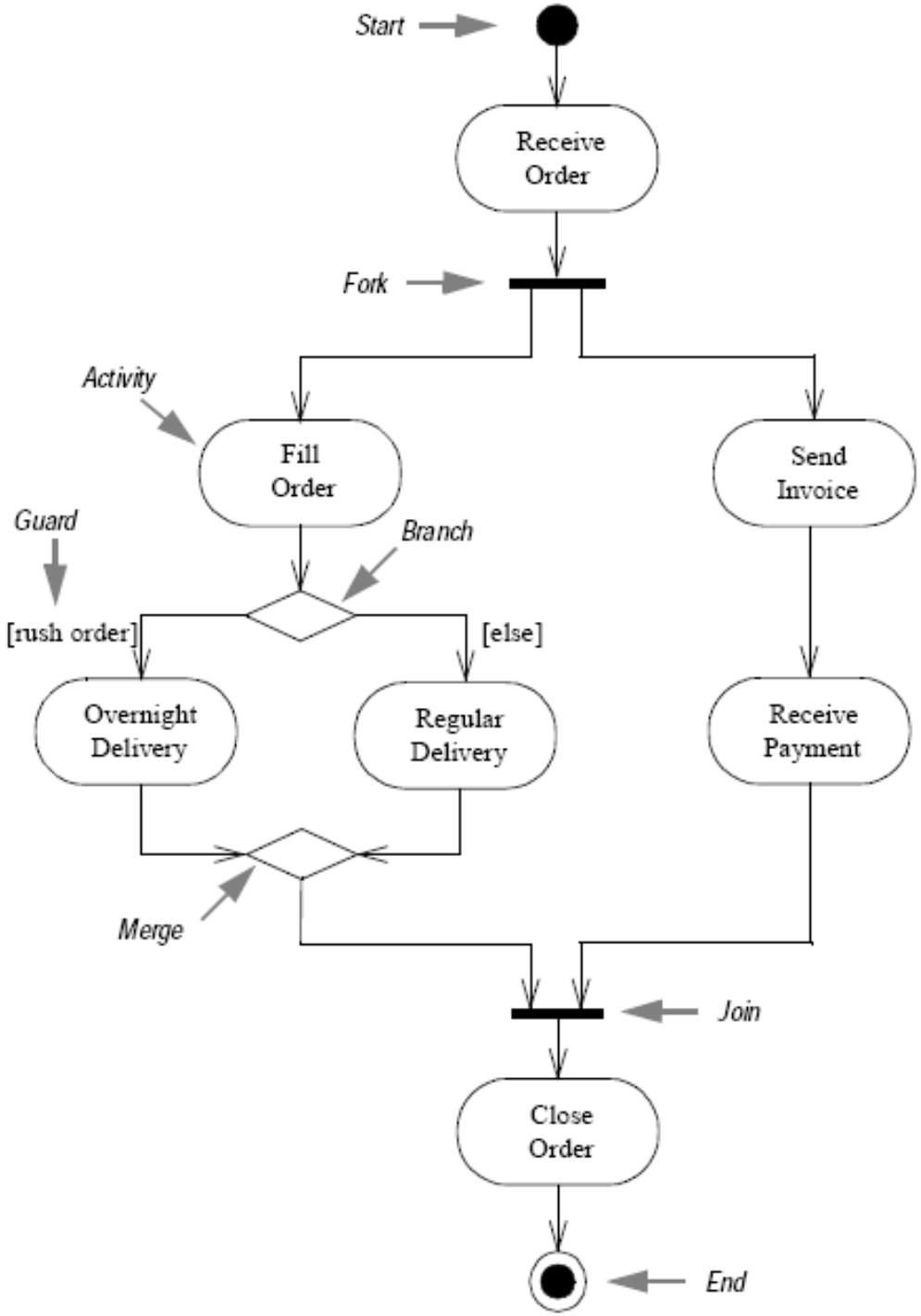
"USE-CASE" İLİŞKİ TÜRLERİ: ÖRNEK



“USE-CASE” MODELİ

- Sistemin tüm “use-case” diyagramları, “use-case” modelini tanımlar
  - Sistem belirli büyüklüğün üzerinde olduğunda, gereksinimler birden fazla “use-case” diyagramı kullanılarak tanımlanır
  - Sistem sınırını gösteren dikdörtgen kutu sistemin içinde ve dışında kalan öğeleri belirtmek için kullanılır

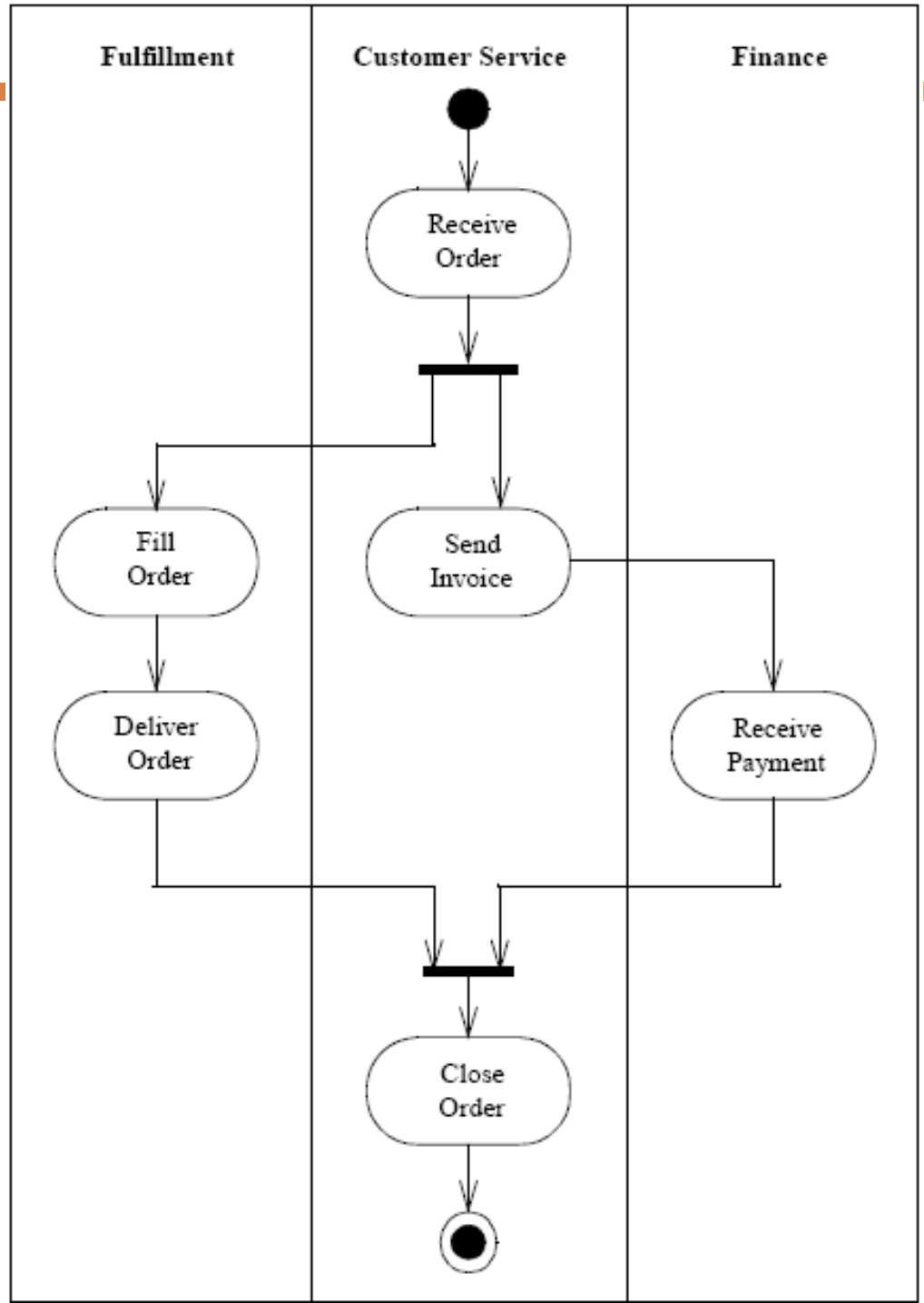
"ACTIVITY DIAGRAM": ÖRNEK



"ACTIVITY DIAGRAM" MODELLEME ÖĞELERİ

- Etkinlik ("activity")
  - Sistem ve aktörler tarafından yapılan işleri ifade etmek için kullanılır
  - "An activity is a state of doing something"
- Geçiş ("transition")
  - Etkinlikler arasındaki geçişleri ifade etmek için kullanılır
  - Geçişin koşulu varsa geçişin üzerine "guard" eklenerek ifade edilir
- "Branch" / "merge"
  - Koşullu davranışı ifade etmek için kullanılır
    - "A branch has a single incoming transitions and several guarded outgoing transitions … only one of the outgoing transitions can be taken, so the guards should be mutuallyexclusive"
    - "A merge has multiple input transitions and a single output – the end of conditionalbehavior"
- "Fork" / "join"
  - Paralel davranışı ifade etmek için kullanılır
    - "A fork has one incoming transition and several outgoing transitions … when the incoming transition is triggerred, all of the outgoing transitions are taken inparallel"
    - "With a join, the outgoing transition is only taken when all the states on the incoming transitions have completed their activity"

"ACTIVITY DIAGRAM WITH SWIMLANES": ÖRNEK



“USE-CASE” ESASLI GEREKSİNİM ANALİZİ

GEREKSİNİM ANALİZİNDE "USE-CASE" YAKLAŞIMI

- Bakış açısı: Sistem, kullanıcısı için "ne" yapacak ?
  - Sistem kapalı bir kutu ("black-box")
  - Sistem-kullanıcı etkileşimi
  - Sistemin dışarıdan görünen davranışı

- İlgilenmediklerimiz:
  - Sistemin iç yapısı
  - Sistem belirlenen davranışı "nasıl" yapacak ?
  - Belirlenen davranış "nasıl" kodlanacak ?

Bu bakış açısı, sistemdeki tüm işlevselliği değil, kullanıcılar için artı değer oluşturacak işlemleri düşünmemizi sağlar

“USE-CASE” NEDİR? (1)

- Yazılım sisteminin kullanıcısına değer döndüren, dışarıdan gözlemlenebilen davranışının bütünüdür
  - **Değer**
    - Sistem “use case” kapsamındaki işleri gerçekleştirdiğinde oluşacak çıktı, sonuç veya durumdur
  - **Gözlemlenebilirlik**
    - Kullanıcı ve sistem arasındaki etkileşimi tanımlar
  - **Bütünlük**:
    - “Use-case” kullanıcının istediği değerin üretilebilmesi için gereken tüm ilişkili adımları içerir

“USE-CASE” NEDİR? (2)

- Her “use-case”in bir amacı vardır
  - Niye belirlendi ? Kullanıcının hangi işlemini gerçekleştirecek ?
  - Hangi davranışı modelliyor ? Ne yapacak ?
  - Ne zaman sonlanacak ? Hangi değeri üretecek ?

- Örnekler:
  - Öğrencinin bir derse kaydolması
  - Muhasebe görevlisinin aylık bordroları hazırlaması
  - Banka müşterisinin ATM’den para çekmesi
  - Bir kişinin asansörü çağırması

GEREKSİNİM ANALİZİNDE YAPISAL YÖNTEM YA DA "USE-CASE" YAKLAŞIMI: ÖRNEK

- Yapısal yöntem:
  - Sistem müşteri kartını kabul eder.
  - Sistemde işlemlerin yapılabilmesi için şifre geçerli olmalıdır.
  - Sistemde para çekme ve para yatırma seçenekleri mevcuttur.
  - Sistem istendiğinde işlemlerin makbuzunu verir.
  - ……
- Use case yaklaşımı:
  - Müşteri kartını yerleştirir
  - Sistem şifreyi sorar
  - Müşteri şifresini girer
  - Şifre doğruysa, sistem para çekme ve para yatırma seçeneklerini sunar
  - Müşteri para çekme işlemini seçer
  - .....

"USE-CASE" ESASLI GEREKSINIM ANALIZI – KAZANÇLAR

- Kullanıcının gereksinimi olmayan özellikleri tanımlamamızı engeller
- Kullanıcının da anlayabileceği şekilde sistemin davranışlarını ve sorumluluklarını tanımlar
  - Kullanıcı ile iletişimi kolaylaştırır
- Kullanıcı arayüzlerinin tasarlanmasını kolaylaştırır
- Kullanıcı kılavuzlarını yazarken başlangıç noktasını oluşturur
- Geliştirme sürecini başlatır ve tüm temel iş adımlarını birbirine bağlar
- Tasarlanacak test durumlarına esas oluşturur

"USE-CASE" ESASLI GEREKSİNİM ANALİZİ – DİKKAT EDİLECEK NOKTALAR

- "Use-case"ler sistemin iç yapısını tanımlamaz (sistem kapalı kutu)
  - Tasarım öğeleri belirsizdir
    - Tipik olarak birden çok tasarım öğesi, bir use case'in işletilmesi için kullanılır
  - Yapısal veya nesne yönelimli yaklaşımlarda kullanılabilmesinin temel nedeni budur

- "Use case"ler sadece işlevsel gereksinimleri, kullanıcı bakış açısıyla tanımlar
  - Sistemin iç davranışına ilişkin gereksinimler, kullanıcı gereksinimleri gerçekleştirilirken daha sonraki adımlarda karşılanır
    - Bu tür gereksinimler sıklıkla, iş kuralı veya tasarım kısıtı olarak "use-case"lerle ilişkilendirilir

"USE-CASE" ESASLI GEREKSINIM ANALIZI:YÖNTEM

1. Aktörleri ve "use-case"leri belirle
   - Amaç: Sisteminin aktörlerini ve "use-case"lerini belirlemek ve üst seviye "use-case" modelini oluşturmak
     - Aktörler belirlenir
     - "Use-case"ler belirlenir
     - Her aktör ve "use case" kısaca tanımlanır
     - Üst seviye "use-case" modeli tanımlanır

2. "Use-case"leri detaylandır
   - Amaç: Belirlenen tüm "use-case"lerin iş akışlarını detaylı olarak tanımlamak
     - Ana akış tanımlanır
     - Alternatif akışlar tanımlanır

"USE-CASE" ESASLI GEREKSİNİM ANALİZİ:YÖNTEM (DEVAMI)

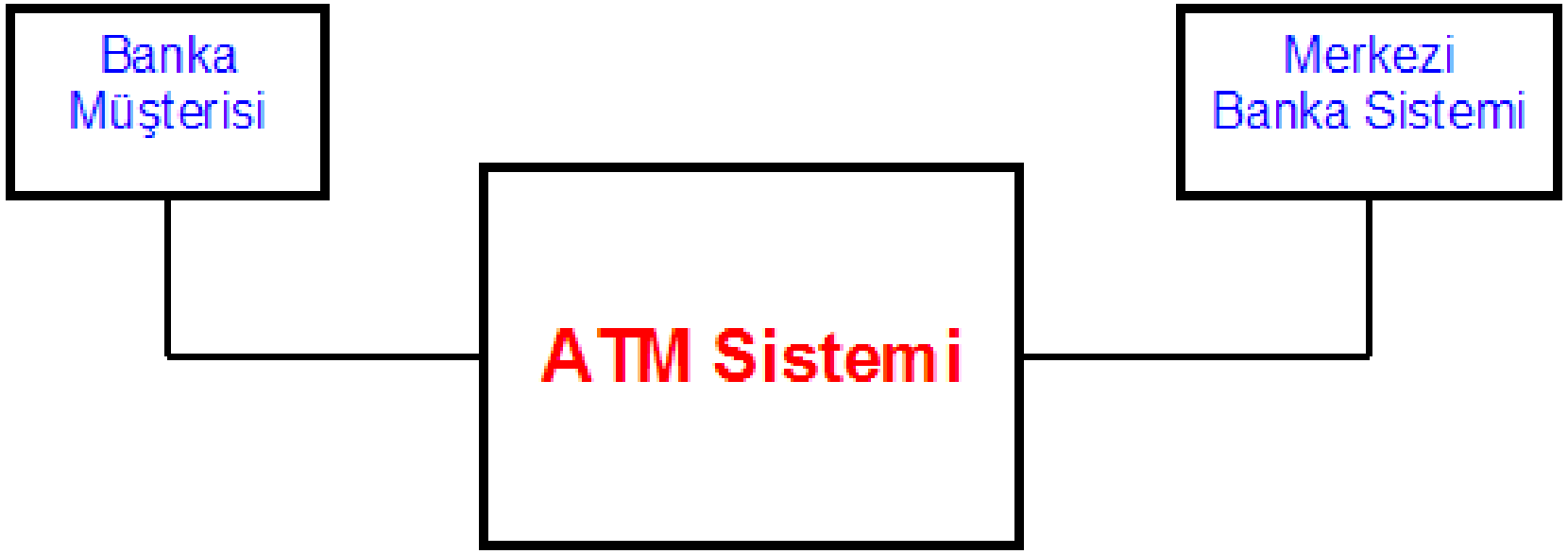
3. "Use-case" modelini yapılandır
   - Amaç: Oluşturulan use case modelini ortak noktaları en aza indirecek şekilde yapılandırmak
     - Gereken yerlerde "extend" ve "include" ilişkileri kullanılabilir
     - Yapılandırılan "use-case" modeli, iş süreçlerini referans alınarak değerlendirilir

4. Kullanıcı arayüzlerini tanımla
   - Amaç: Use case tanımları esas alınarak kullanıcı arayüzlerini üst seviyeli olarak tanımlamak
     - Kağıt üzerinde çizim yapılabilir
     - Arayüz prototipleme aracı kullanılabilir

ÖRNEK: ATM UYGULAMASI

- 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 UYGULAMASI – KAPSAM



Bağlam (“context”) diyagramı

ATM UYGULAMASI (ADIM 1. AKTÖRLERİ VE "USE CASE"LERİ BELİRLE) – AKTÖRLER

- Soru: ATM uygulama yazılımının kullanıcıları kimlerdir?
  - Banka müşterisi
  - Merkezi Banka Sistemi

ATM UYGULAMASI (ADIM 1. AKTÖRLERİ VE "USE CASE"LERİ BELİRLE) – "USE CASE"LER

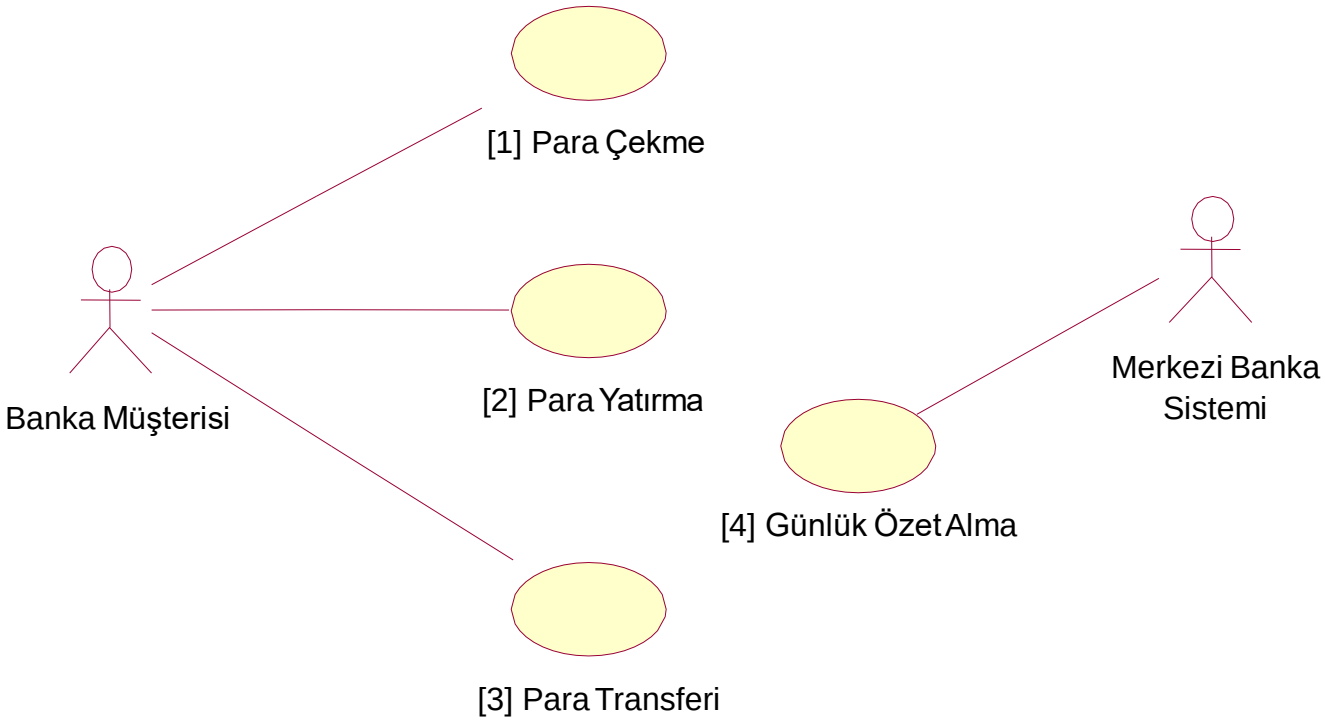
- Soru: Belirlenen aktörler ATM'den ne istiyorlar ?
  - Aktör: Banka müşterisi
    - Para çekme
    - Para yatırma
    - Para transferi
  - Aktör: Merkezi Banka Sistemi
    - Günlük özet alma

ATM UYGULAMASI (ADIM 1. AKTÖRLERİ VE "USE CASE"LERİ BELİRLE) – KISA TANIMLAR

- Aktör: Banka müşterisi
  - Bankada hesabı ve banka kartı olan, ATM'den işlem yapma hakkı olan kişi

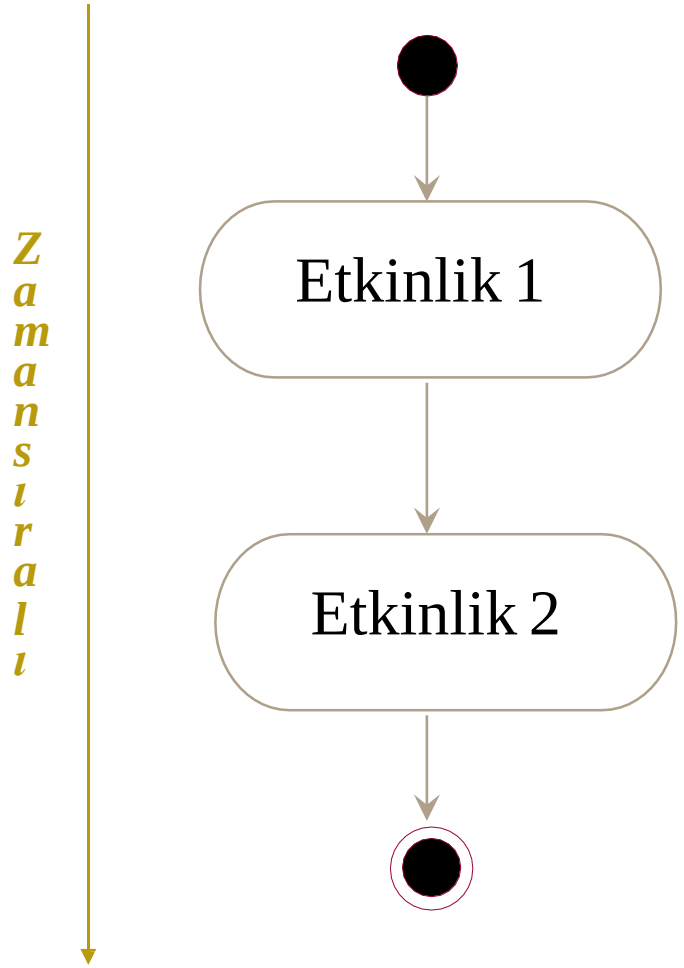
- Use case: Para çekme
  - Banka müşterisinin nasıl para çekeceğini tanımlar. Para çekme işlemi sırasında banka müşterisinin istediği tutarı belirtmesi ve hesabında bu tutarın mevcut olması gerekir.

ATM UYGULAMASI (ADIM 1. AKTÖRLERİ VE "USE CASE"LERİ BELİRLE) – "USE-CASE" DİYAGRAMI



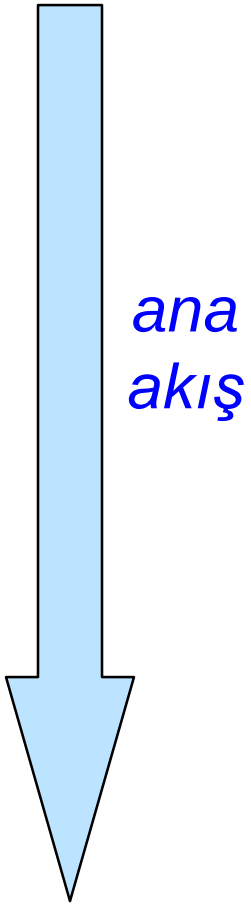
Her use case biricik ("unique") olarak numaralandırılmış

“USE CASE” DETAYI: ETKINLIK ZINCIRI



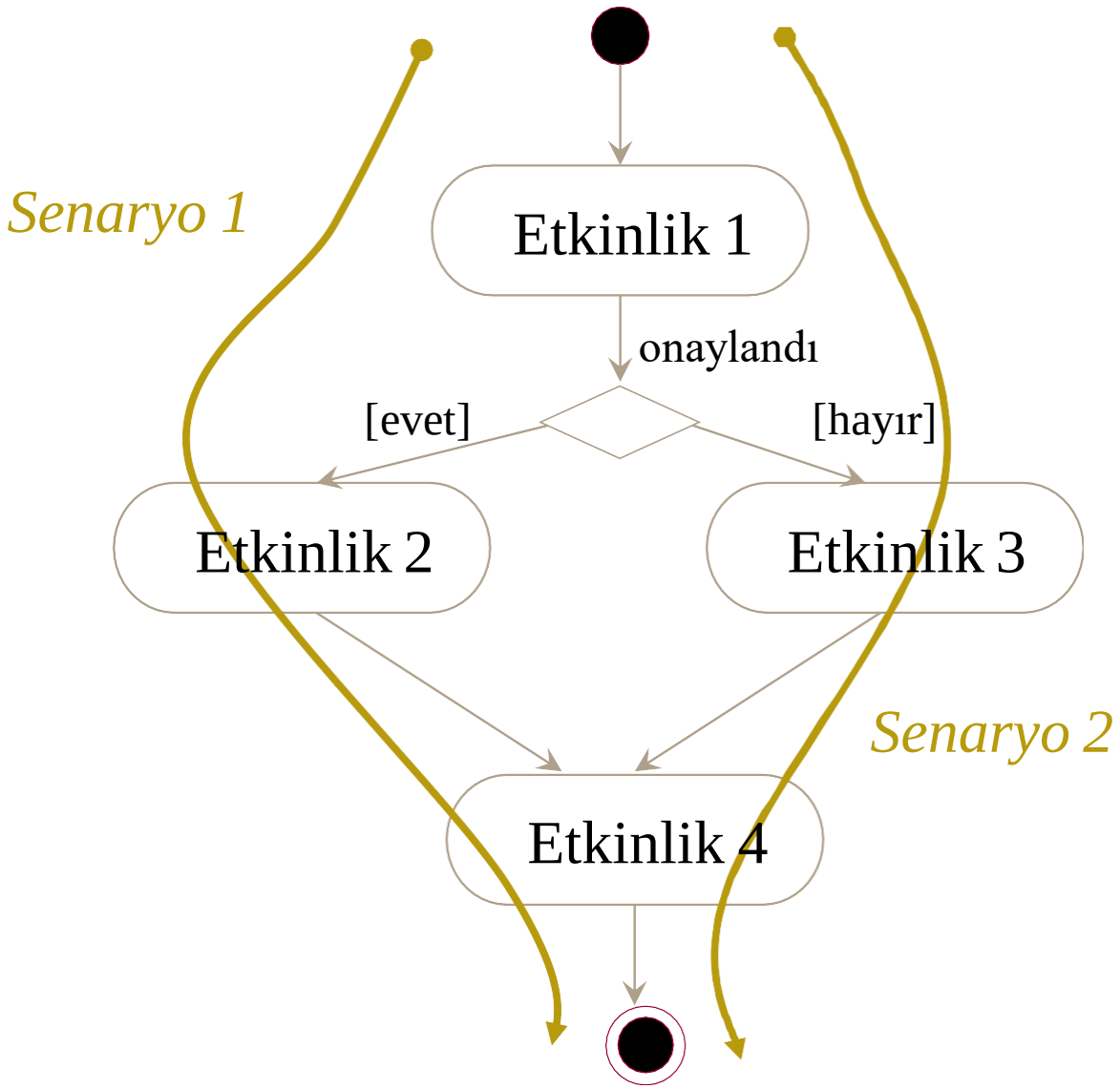
ATM Uygulaması (Adım 2. “Use Case”leri Detaylandır) – Para Çekme (Ana Akış)

1. Banka Müşterisi kartını yerleştirir
2. ATM kartı okur
3. Banka Müşterisi şifreyi girer
4. ATM işlem seçeneklerini gösterir
5. Banka Müşterisi “para çekme” işlemini seçer
6. ATM olası para tutarlarını gösterir
7. Banka Müşterisi para tutarını girer
8. ATM para çekme talebini Merkezi Banka Sistemi’ne iletir
9. Merkezi Banka Sistemi, Banka Müşterisi’nin hesabını kontrol eder
10. Merkezi Banka Sistemi, Banka Müşterisi’nin hesabından tutarı düştü ve işlem sonucunu ATM’ye iletir
11. ATM nakit parayı verir
12. ATM kartı iade eder

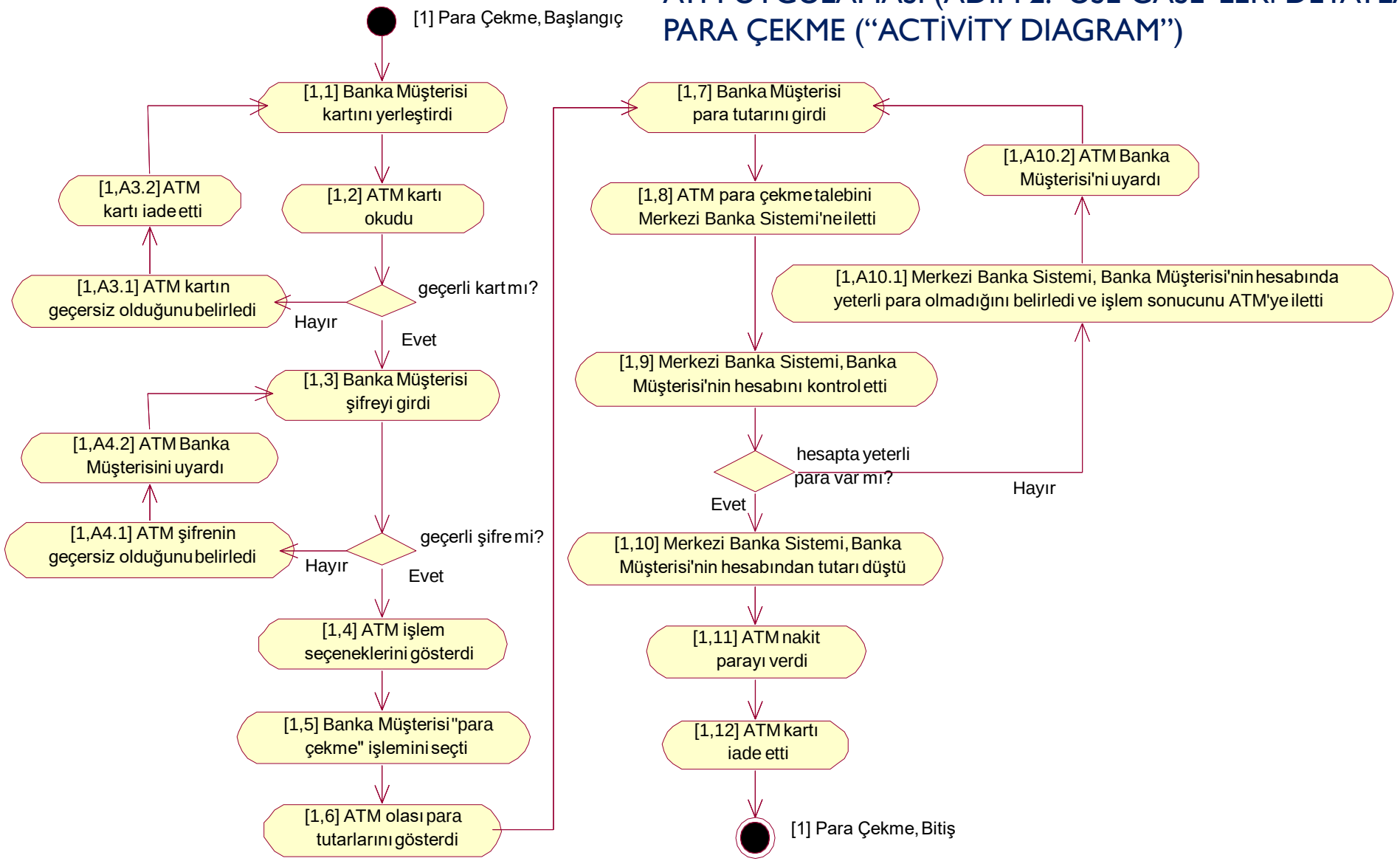


Olası sapma noktaları

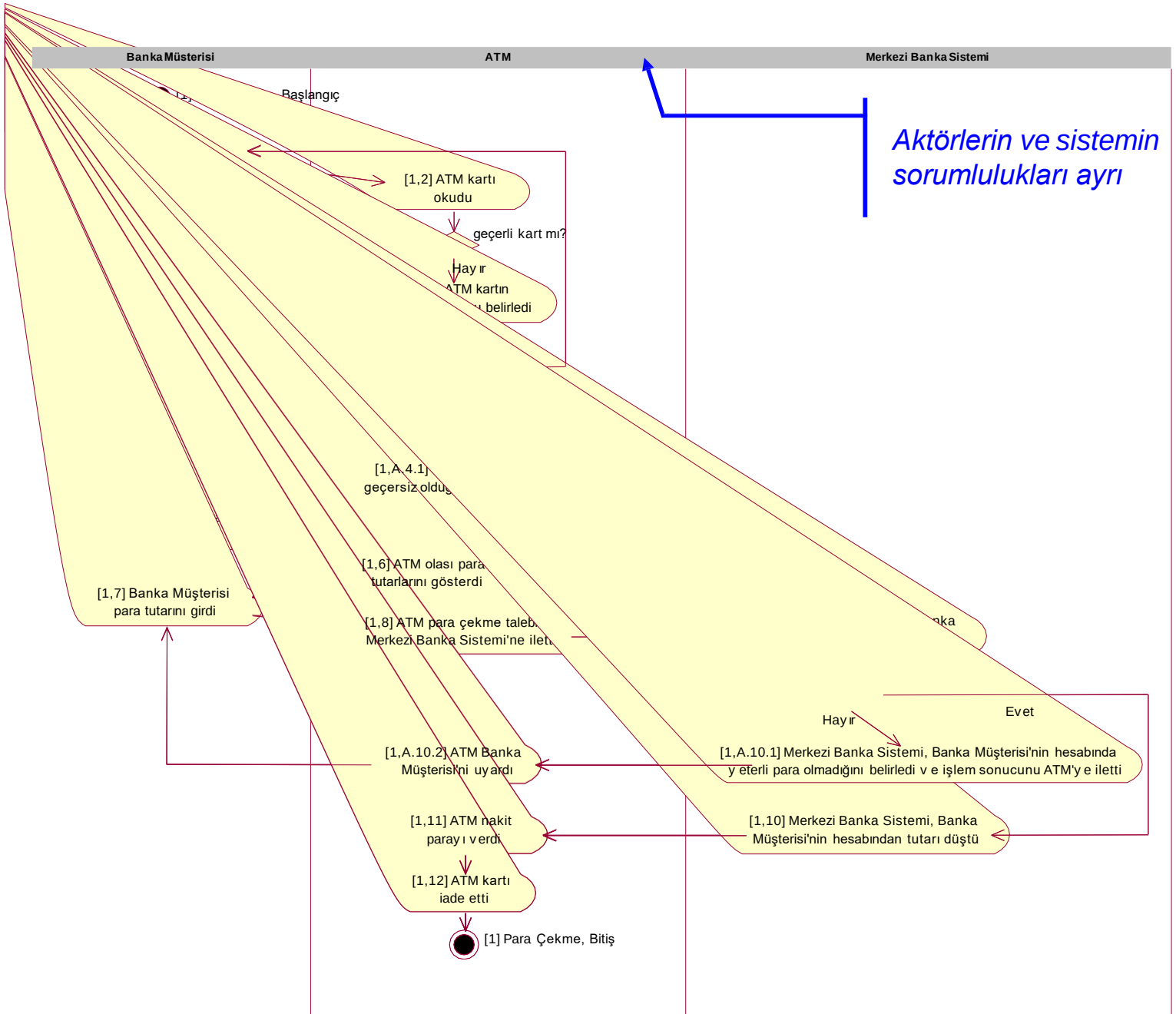
“USE CASE” DETAYI: ALTERNATIF AKIŞLAR



ATM UYGULAMASI (ADIM 2."USE CASE"LERİ DETAYLANDIR) – PARA ÇEKME ("ACTİVİTY DIAGRAM")



PARA ÇEKME – "ACTIVITY DIAGRAM WITH SWIMLANES"

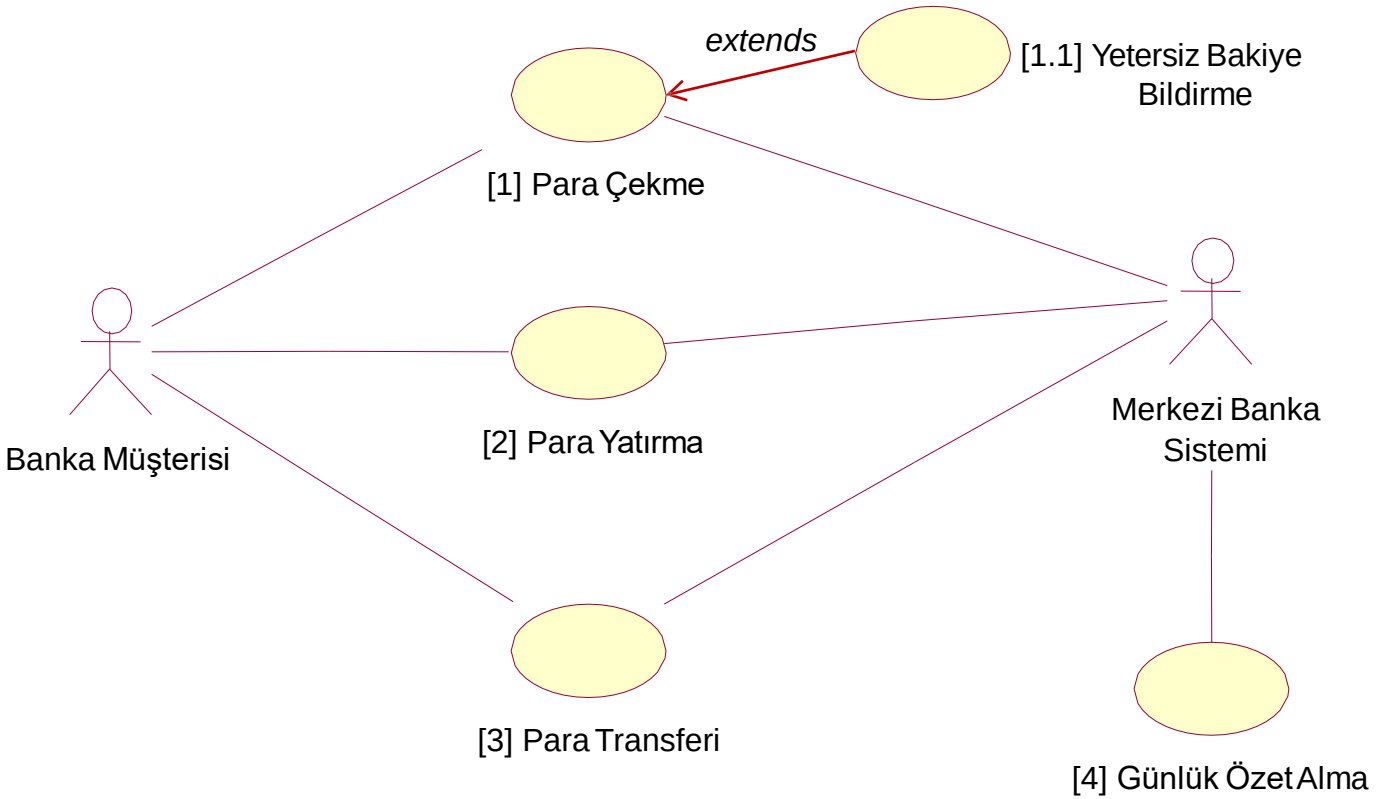


PARA ÇEKME: ALTERNATİF TANIM

Use Case No:	12		
Use Case Adı:	Para Çekme		
Tanımlayan:		Son Değiştiren:	
Tanımlama Tarihi:		Son Değişiklik Tarihi:	

Aktör:	Banka müşterisi, Merkezi Banka Sistemi
Kısa Tanımı:	Banka müşterisinin nasıl para çekeceğini tanımlar.
Önkoşul:	Merkezi Banka Sistemi erişilebilir durumdadır.
Sonkoşul:	Banka müşterisi nakit parayı ve banka kartını alır.
Önceliği:	1
Kullanım sıklığı:	Çok sık
Ana akış:	1. Banka Müşterisi kartını yerleştirdi 2. ATM kartı okudu 3. Banka Müşterisi şifreyi girdi 4. ATM işlem seçeneklerini gösterdi 5. Banka Müşterisi "para çekme" işlemini seçti 6. ATM olası para tutarlarını gösterdi 7. Banka Müşterisi para tutarını girdi 8. ATM para çekme talebini Merkezi Banka Sistemi'ne iletti 9. Merkezi Banka Sistemi, Banka Müşterisi'nin hesabını kontrol etti 10.Merkezi Banka Sistemi, Banka Müşterisi'nin hesabından tutarı düştü 10. ve işlem sonucunu ATM'ye iletti 11. ATM nakit parayı verdi 12. ATM kartı iade etti
Alternatif Akış:	A3: ATM kartın geçersiz olduğunu belirledi A3.1. ATM kartı iade etti A3.2. *Adım-1'den devam et* A4: ATM şifrenin geçersiz olduğunu belirledi A4.1. ATM Banka Müşterisi'ni uyardı A4.2. *Adım-3'ten devam et* A10: Merkezi Banka Sistemi, Banka Müşterisi'nin hesabında yeterli para olmadığını belirledi ve işlem sonucunu ATM'ye iletti A10.1. ATM Banka Müşterisi'ni uyardı A10.2. *Adım-7'den devam et*
İçerdiği use case'ler:	
Özel gereksinimler:	(Adım-2) ATM kartı 3 saniye içinde okumalıdır.
Varsayımlar:	
Not:	

ATM UYGULAMASI (ADIM 3. "USE-CASE" MODELİNİ YAPILANDIR) – "USE-CASE" DİYAGRAMI



ATM UYGULAMASI (ADIM 4. KULLANICI ARAYÜZLERİNİ TANIMLA) – PARA ÇEKME "USE CASE"İ İÇİN KULLANICI ARAYÜZÜ

(Para Çekme ana akışı 5, 6 ve 7 no'lu adımlarıyla ilişkili kullanıcı arayüzleridir.)

“USE-CASE” ESASLI GEREKSİNİM ANALİZİ: BAŞARI İÇİN ANAHTAR NOKTALAR

- “Use-case” modelini iteratif olarak geliştirin
- Kullanıcıları analize dahil edin
- “Use-case”leri görsel olarak modelleyin
- “Use-case”leri işlevsel olmayan gereksinimleri çıkarmak için kullanın
- “Use-case”leri ve “use-case” senaryolarını önceliklendirin
- “Use-case”leri doğrulayın ve diğer geliştirme öğelerine izlenirliğini kurun

SINIF ÇALIŞMASI: KÜTÜPHANE DESTEK SİSTEMİ

- Kütüphane işlemlerini desteklemek amacıyla bir yazılım sistemi oluşturulacaktır.
- Sistem; kayıtlı müşterilere, yine kayıtlı kitap ve dergileri ödünç verecektir.
- Kütüphane, yeni başlıklı kitap ve dergilerin satın almasını yapacaktır. Popüler başlıklar, birden çok kopya satın alınacaktır. Eski kitap ve dergiler, zaman aşımına uğradıklarında veya çok yıprandıklarında yok edilecektir.
- Kütüphanede, müşterilerle iletişimi sağlayacak ve yaptığı işler sistem tarafından desteklenecek bir kütüphane görevlisi bulunacaktır.
- Müşteri, kütüphanede o anda bulunmayan bir kitap veya dergiyi rezerve edebilecektir. Kitap veya dergi kütüphaneye geri döndürüldüğünde, rezervasyonu yapan müşteri haberdar edilecektir. Rezervasyon, müşteri kitap veya dergiyi ödünç aldığında veya müşterinin özel isteği üzerine iptal edilecektir.
- Sistem; kitap ve dergi başlıklarının, kitap ve dergi kopyalarının, müşterilerin, ödünç işlemlerinin ve rezervasyonların kaydedilmesine, güncellenmesine ve silinmesine olanak sağlayacaktır.
- Sistem tüm popüler bilgisayar ortamlarında (UNIX, Windows, OS/2, vb.) çalışacak ve modern bir kullanıcı ara yüzüne sahip olacaktır.
- Sistem yeni işlevler eklemek suretiyle genişletilebilir olacaktır.

REFEREANSLAR

- Software Engineering (10th. Ed.); Ian Sommerville; 2015.
- Guide to Software Engineering Body of Knowledge (v3); 2014.
- Hacettepe Üniversitesi BBS-651, A. Tarhan, 2019.