İçindekiler

NESNE AMBARI

1. Projenin Tanımı

Birden fazla fakülteye, enstitüye, yüksekokula ve meslek yüksekokuluna sahip, sahip olduğu fakülteler, enstitüler, yüksekokullar ve meslek yüksekokulları bir veya birden daha fazla bölüme sahip bir üniversitenin, sayılan bütün bu birimlerinde gösterilmekte olan derslerle ilgili nesnelerin (geçmiş senenin ya da senelerin sınav soruları, ders notları, animasyon, video, fotoğraf vs.) üye olan kullanıcılar (öğrenci, öğretim üyesi) tarafından sisteme yüklenip, saklandığı, ilgili nesnenin ait olduğu dersin sorumlusu tarafından onaylanıp yayınlandığı ve üye olmayan kullanıcılar dahil herkes tarafından görüntülendiği, istenildiğinde üye olan kullanıcılar tarafından indirilebilindiği web tabanlı bir nesne ambarı uygulaması.

2. Başlangıç Aşamasında Olabilecek Nesneler:

Fakülte	nesne
Enstitü	nesne
Yüksekokul	nesne
Meslek yüksekokul	nesne
Üniversite	nesne
Ders	nesne
Nesne	nesne
Sınav soruları	nesne
Ders notları	nesne
Animasyon	nesne
Video	nesne
Fotoğraf	nesne
Bölüm	nesne
Üye olan kullanıcı	nesne
Öğrenci	nesne
Öğretim üyesi	nesne
Üye olmayan kullanıcı	nesne

3. Projenin Amaçları

Farklı formattaki materyallerin bir platformda toplanması ve kullanıcıların ihtiyaçları doğrultusunda onlara sunulması sonucunda aşağıdaki amaçlar hedeflenmektedir:

- Öğrencileri ilgilendikleri alanlarda bilgilendirmek.
- Öğrencilerin derslerinde başarılı olmalarını sağlamak.
- Anlaşılması güç olan konularda, olumlu etkisi çok yüksek olan teknoloji yardımıyla öğrencilerin konunun özünü kavramasını sağlamak.
- Öğrenci ve öğretim üyelerinin dilerlerse ders materyallerini paylaşmasını kolaylaştırmak.

4. Senaryolar

1. Üye kaydı:

Ön koşul: Kullanıcı, kayıt ol arayüzünü açmış olmalı.

- Kullanıcı, bilgilerini (adını, soyadını, elektronik posta adresini, parolasını, parolasının tekrarını, öğrenci numarasını, öğrenim düzeyini (ön lisans, lisans, yüksek lisans, doktora), fakülte / enstitü / yüksekokul / meslek yüksekokulunu, bölümünü) girer.
- İşlemler onaylanır.

İstinası: Kullanıcı hatalı elektronik posta adresi girmiştir.
Kullanıcı veri tabanında kayıtlıdır.

2. **Üye girişi:**

Ön koşul: Kullanıcı, kullanıcı adı ve şifreye sahip olmalı.

- Kullanıcı bilgilerini (elektronik posta adresini, parolasını) girer.
- İşlemler onaylanır.

İstisnası: Girilen elektronik posta adresi veya parola veri tabanında kayıtlı değildir.

3. **Nesne listeleme:**

Ön koşul: Kullanıcı, nesne listeleme arayüzünü açmış olmalı.

- Kullanıcı; belirlediği arama kriterlerine göre (menülerden veya detaylı arama ekranından; fakülte / enstitü / yüksekokul / meslek yüksekokul, bölüm, ders, materyal türü (text, video, animasyon, resim vs.), anahtar kelimeler), görüntülemek istediği nesneyi seçer.
- İşlemler onaylanır.

İstisnası: Arama kriterlerine uygun nesne bulunamamıştır.

4. **Nesne indirme:**

Ön koşul: Kullanıcı, üye girişi yapmış olmalı.

- Kullanıcı aradığı nesneyi listeler.
- Kullanıcı indirmek istediği nesneyi seçer.
- İşlemler onaylanır.

İstisnası: Arama kriterlerine uygun nesne bulunamamıştır.

5. **Nesne yükleme:**

Ön koşul: Kullanıcı bu işlemi yapma yetkisine sahip olmalı.

- Kullanıcı, yükleyeceği nesnenin, bilgilerini (fakülte / enstitü / yüksekokul / meslek yüksekokul, bölüm, ders, nesne türü) seçer.
- Kullanıcı, nesne hakkındaki bilgileri(nesne başlığı, nesne açıklaması ve anahtar kelime) girer.
- Kullanıcı, kendi bilgisayarında bulunan, yüklemek istediği nesneyi seçer.
- İşlemler onaylanır.

İstisnası: Veri tabanında aynı nesne vardır.

6. **Nesne güncelleme:**

Ön koşul: Kullanıcı bu işlemi yapma yetkisine sahip olmalı.
Kullanıcı nesneyi kendi yüklemiş olmalı.

- Kullanıcı yüklediği bütün nesneleri listeler.
- Kullanıcı, güncellemek istediği nesneyi seçer.
- Kullanıcı, nesnede gerekli değişiklikleri yapar ve işlemler onaylanır.

İstisnası: İstisnası yoktur.

7. Nesne silme:

Ön koşul: Kullanıcı üye girişi yapmış olmalı.
Kullanıcı nesneyi kendi yüklemiş olmalı.

- Kullanıcı yüklediği bütün nesneleri listeler.
- Kullanıcı, silmek istediği nesneyi seçer.
- İşlemler onaylanır.

İstisnası: İstisnası yoktur.

8. Şifre gönderme:

Ön koşul: Kullanıcının sistemde kayıtlı ve geçerli bir elektronik posta adresi olmalı.

- Kullanıcı elektronik posta adresini girer.
- İşlemler onaylanır.

İstisnası: İstisnası yoktur.

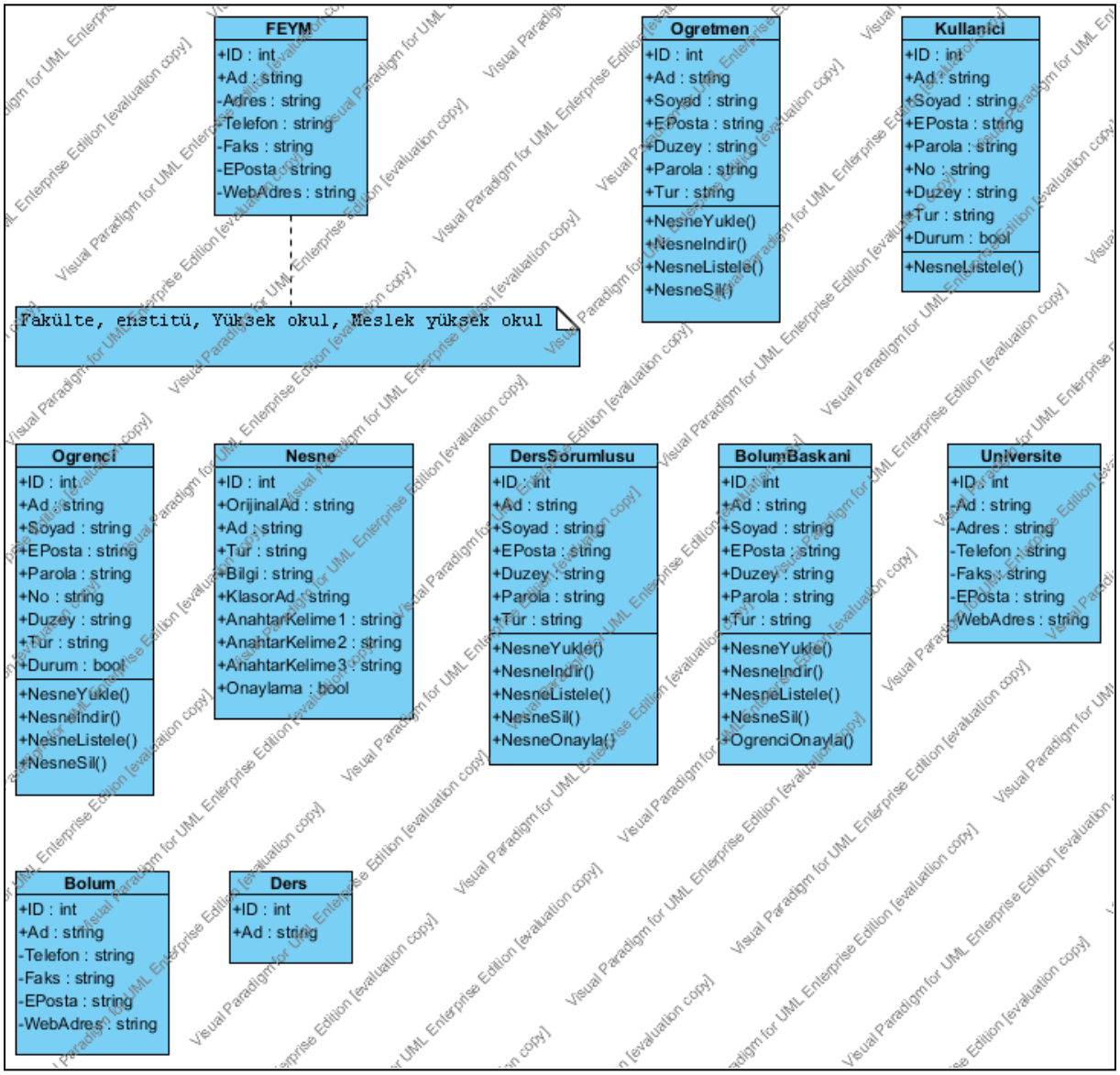
9. Onaylama:

Ön koşul: Nesne veya kullanıcı veri tabanında kayıtlı olmalı.
Ders sorumlusu sisteme girmiş olmalı.

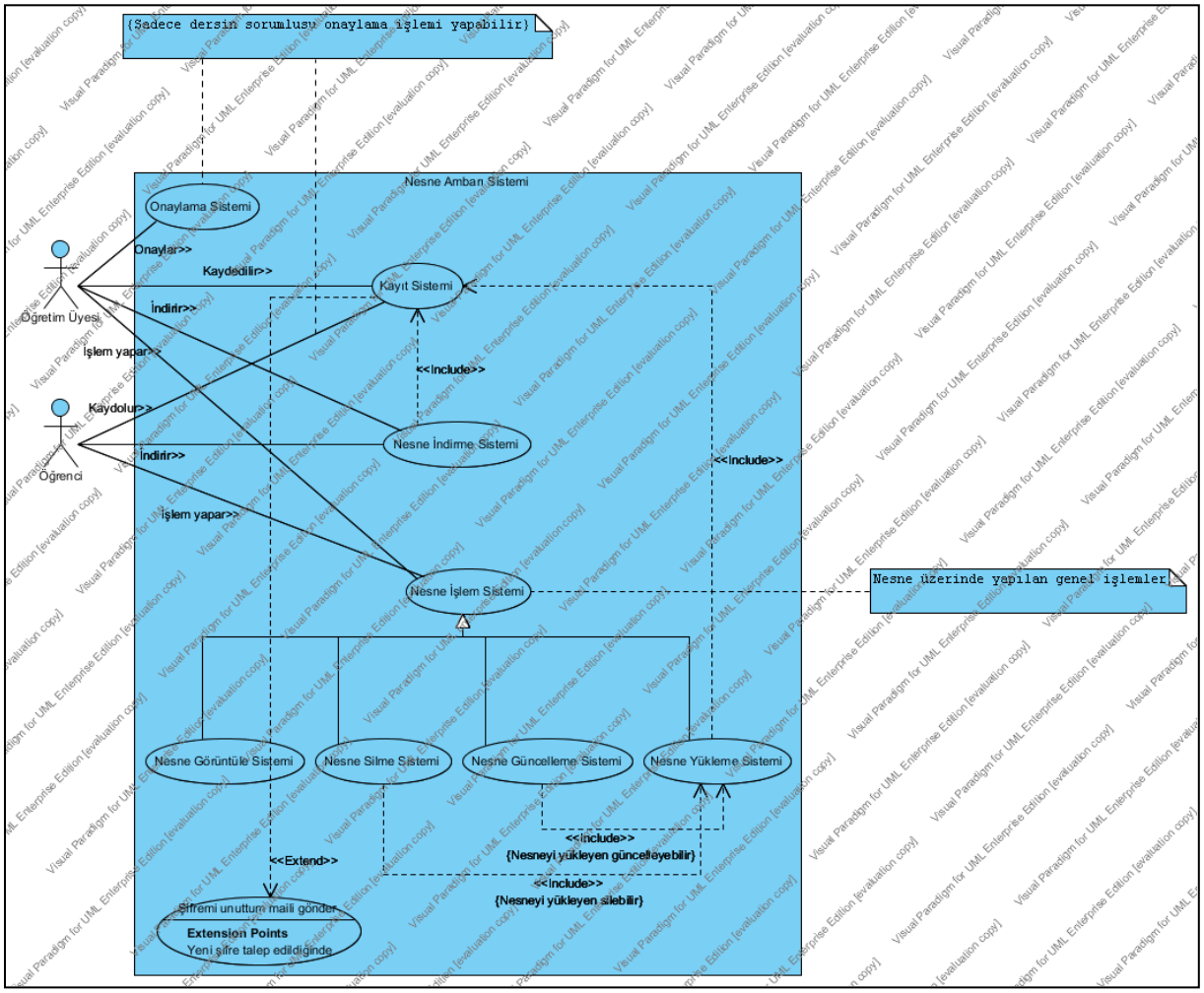
- Nesnenin veya kullanıcının ilgili olduğu dersin sorumlusuna nesnenin yüklendiğine veya kullanıcının kayıt olduğuna dair bilgi e-postası gönderilir.
- Dersin sorumlusu nesneyi veya kullanıcıyı onaylar veya onaylamaz.

İstisnası: İstisnası yoktur.

5. Sınıflar

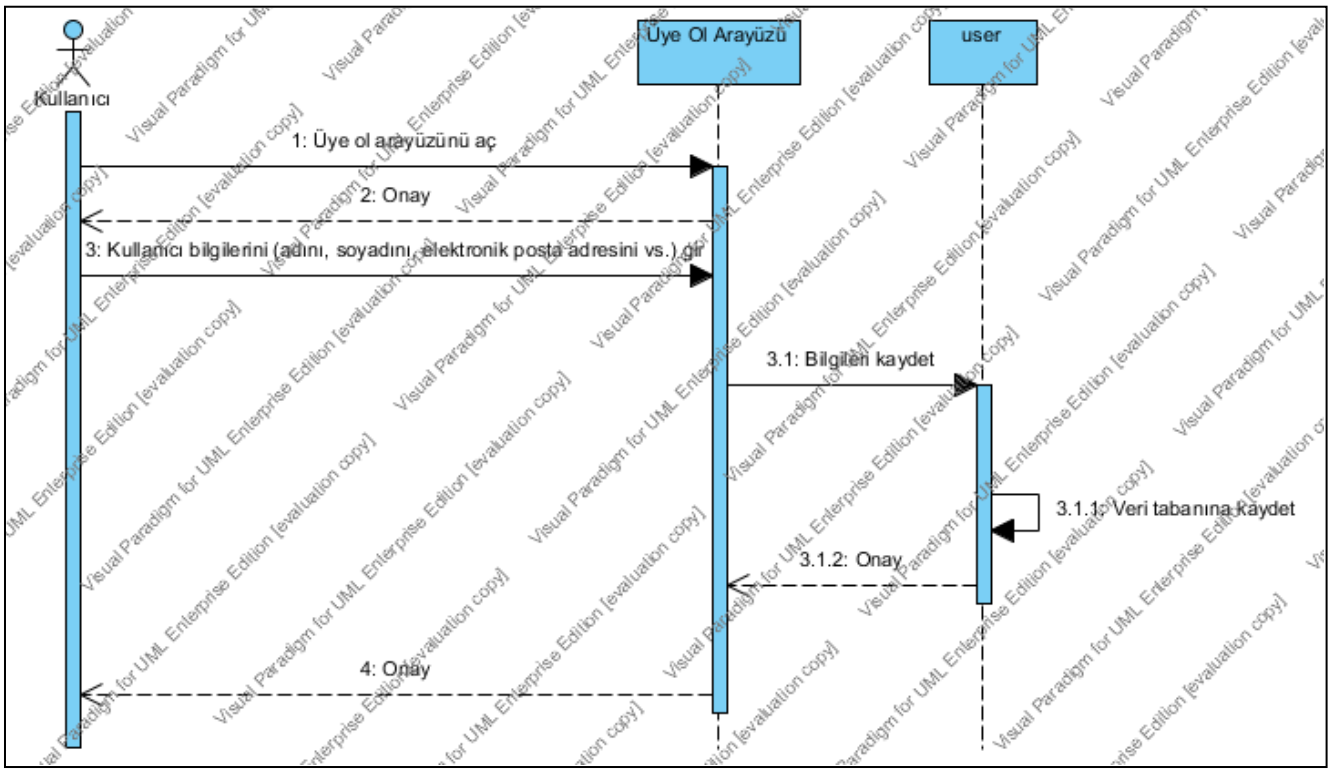

FEYM
+ID : int
+Ad : string
-Adres : string
-Telefon : string
-Faks : string
-EPosta : string
-WebAdres : string
Fakülte, enstitü, Yüksek okul, Meslek yüksek okul
Ogretmen
+ID : int
+Ad : string
+Soyad : string
+EPosta : string
+Duzey : string
+Parola : string
+Tur : string
+NesneYukle()
+NesneIndir()
+NesneListele()
+NesneSil()
Kullanici
+ID : int
+Ad : string
+Soyad : string
+EPosta : string
+Parola : string
+No : string
+Duzey : string
+Tur : string
+Durum : bool
+NesneListele()
Ogrenci
+ID : int
+Ad : string
+Soyad : string
+EPosta : string
+Parola : string
+No : string
+Duzey : string
+Tur : string
+Durum : bool
+NesneYukle()
+NesneIndir()
+NesneListele()
+NesneSil()
Nesne
+ID : int
+OrijinalAd : string
+Ad : string
+Tur : string
+Bilgi : string
+KlasorAd : string
+AnahtarKelime1 : string
+AnahtarKelime2 : string
+AnahtarKelime3 : string
+Onaylama : bool
DersSorumlusu
+ID : int
+Ad : string
+Soyad : string
+EPosta : string
+Duzey : string
+Parola : string
+Tur : string
+NesneYukle()
+NesneIndir()
+NesneListele()
+NesneSil()
+NesneOnayla()
BolumBaskani
+ID : int
+Ad : string
+Soyad : string
+EPosta : string
+Duzey : string
+Parola : string
+Tur : string
+NesneYukle()
+NesneIndir()
+NesneListele()
+NesneSil()
+OgrenciOnayla()
Universite
+ID : int
-Ad : string
-Adres : string
-Telefon : string
-Faks : string
-EPosta : string
-WebAdres : string
Bolum
+ID : int
+Ad : string
-Telefon : string
-Faks : string
-EPosta : string
-WebAdres : string
Ders
+ID : int
+Ad : string

6. Kullanım Durumu Diyagramı

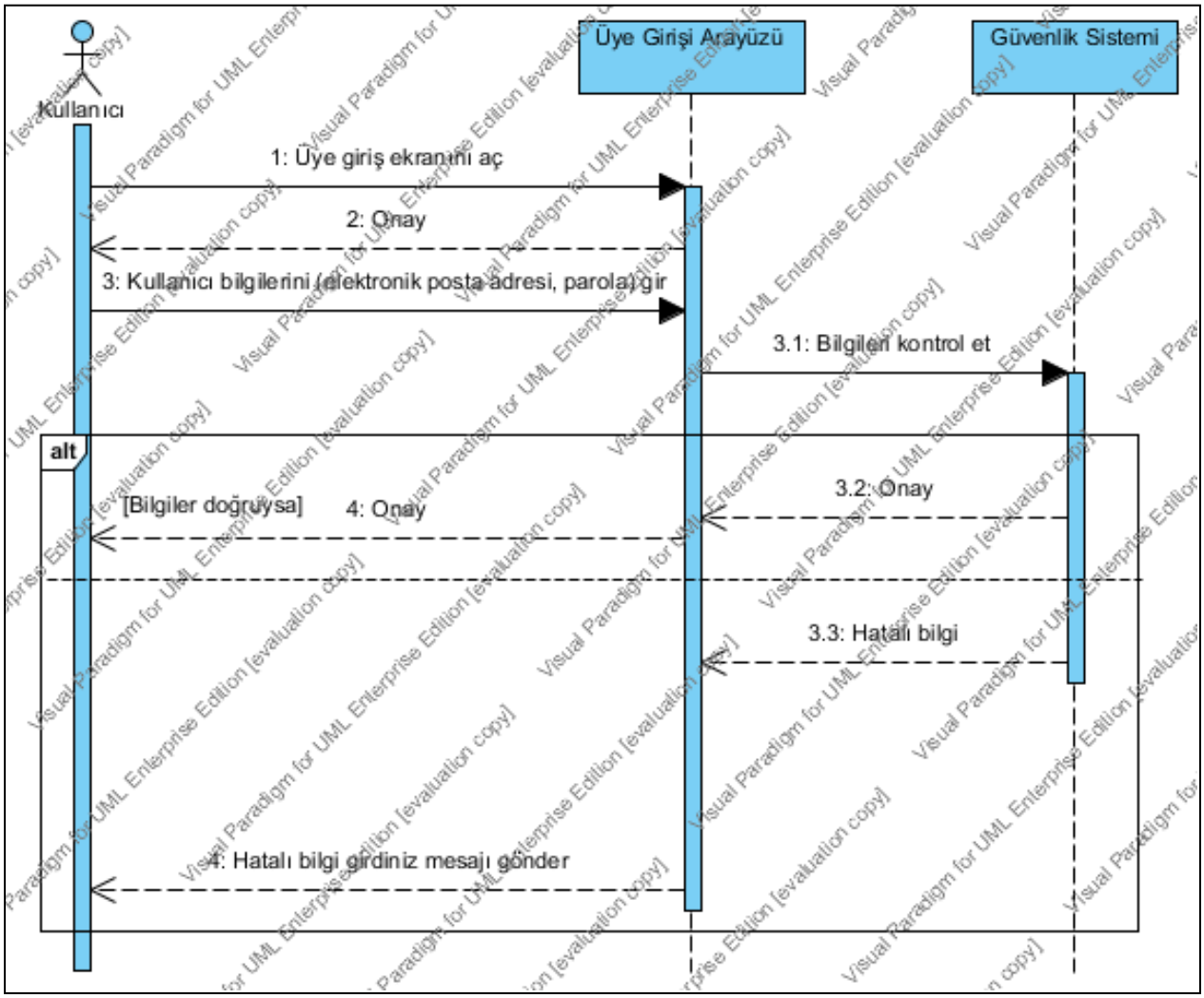


7. Ardışıklık Diyagramları

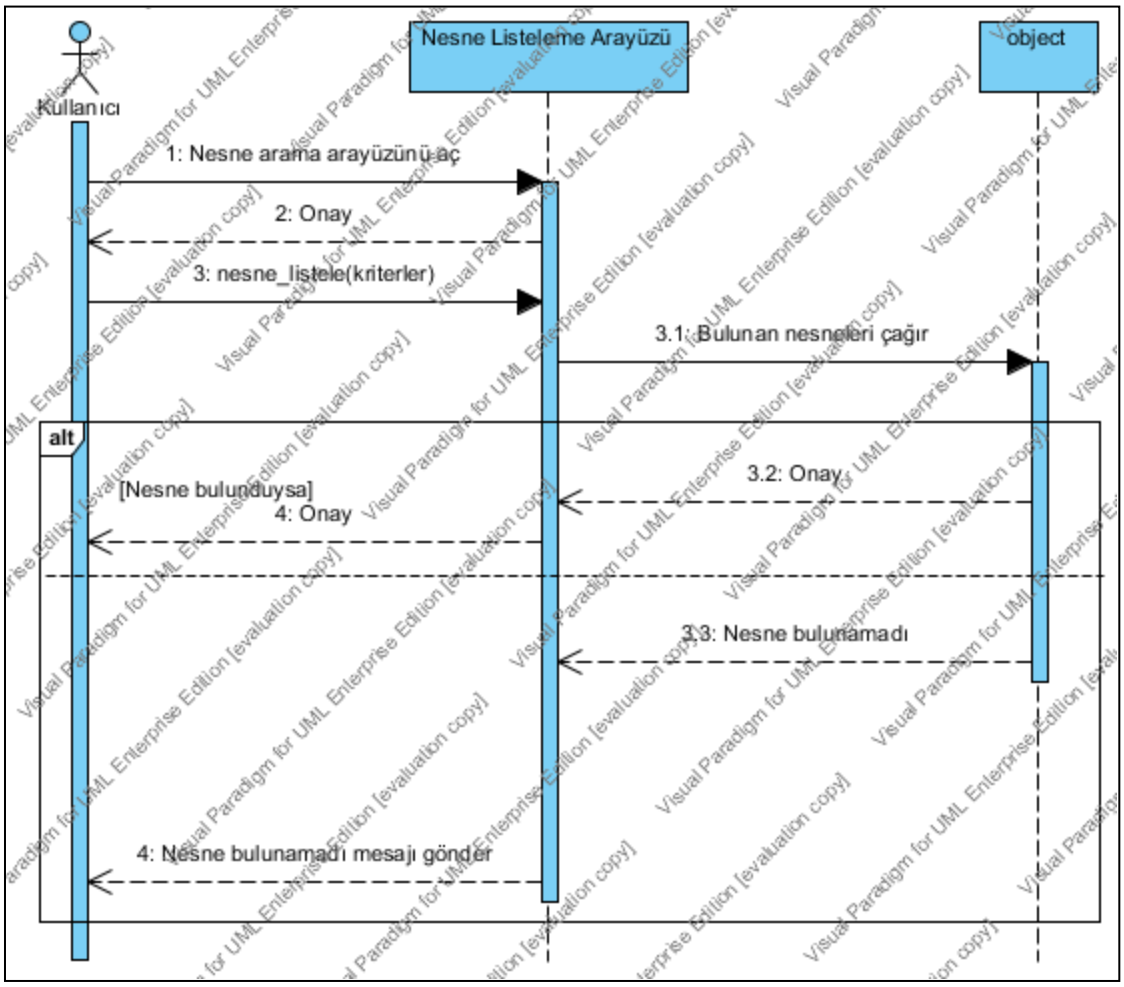
Senaryo 1: Üye Kaydı



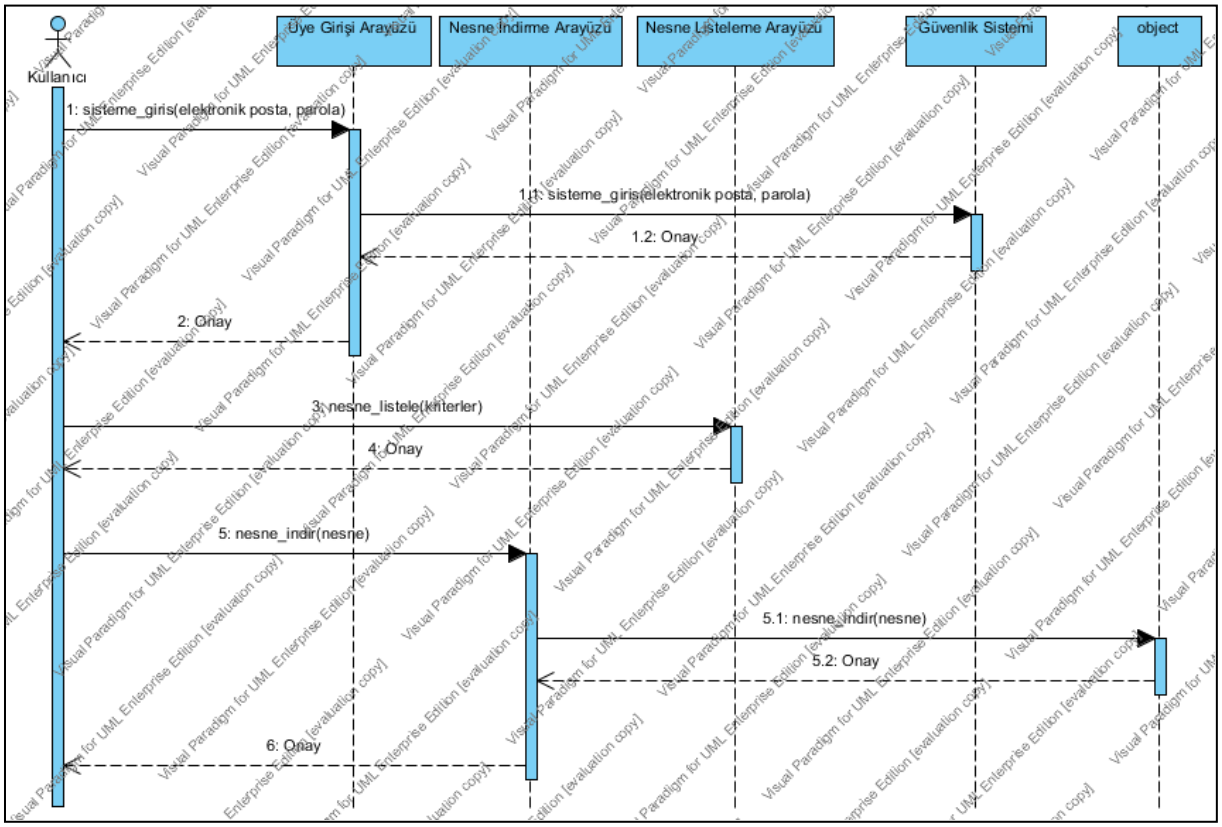
Senaryo 2: Üye Girişi



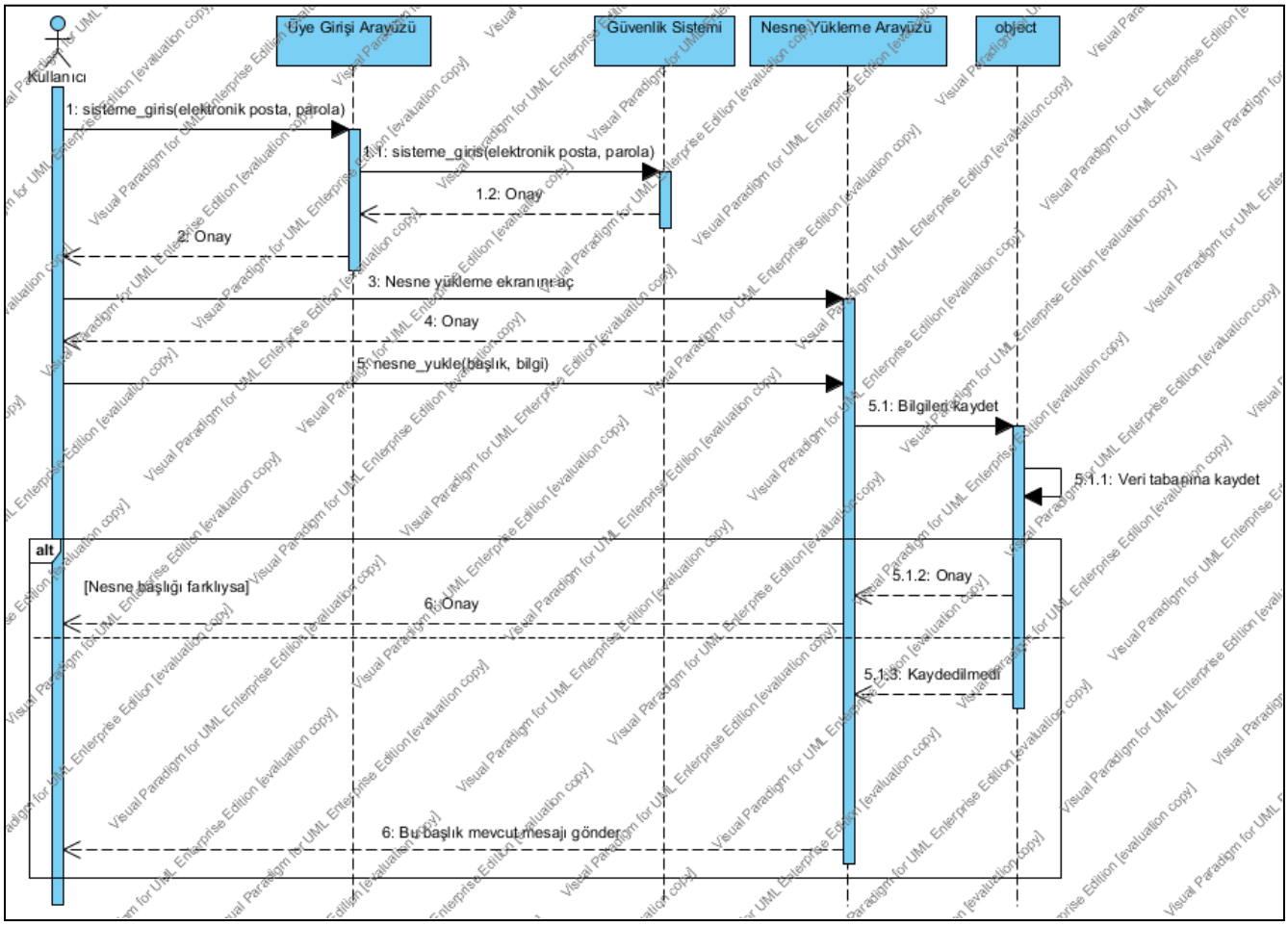
Senaryo 3: Nesne Listeleme



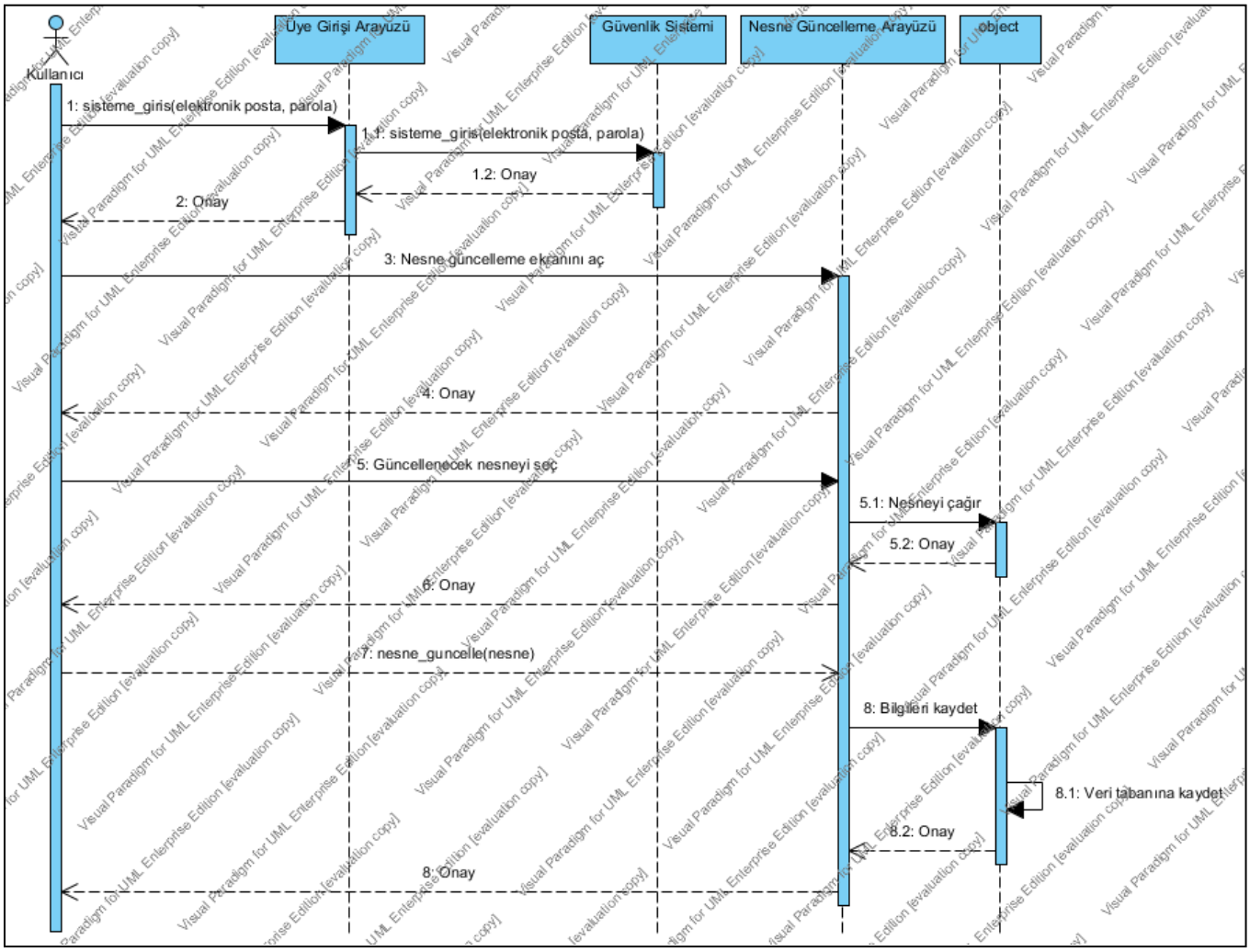
Senaryo 4: Nesne İndirme



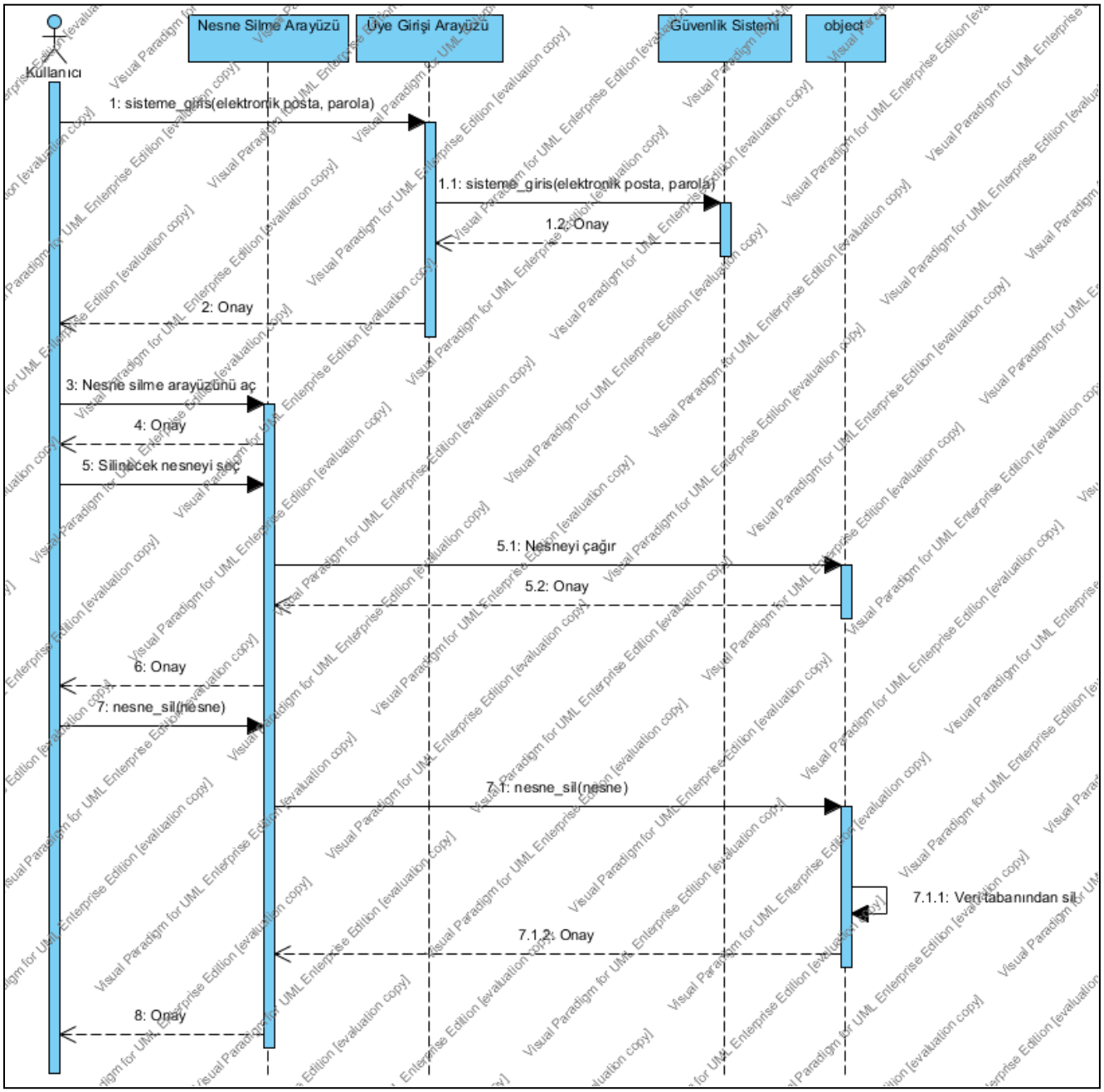
Senaryo 5: Nesne Yükleme



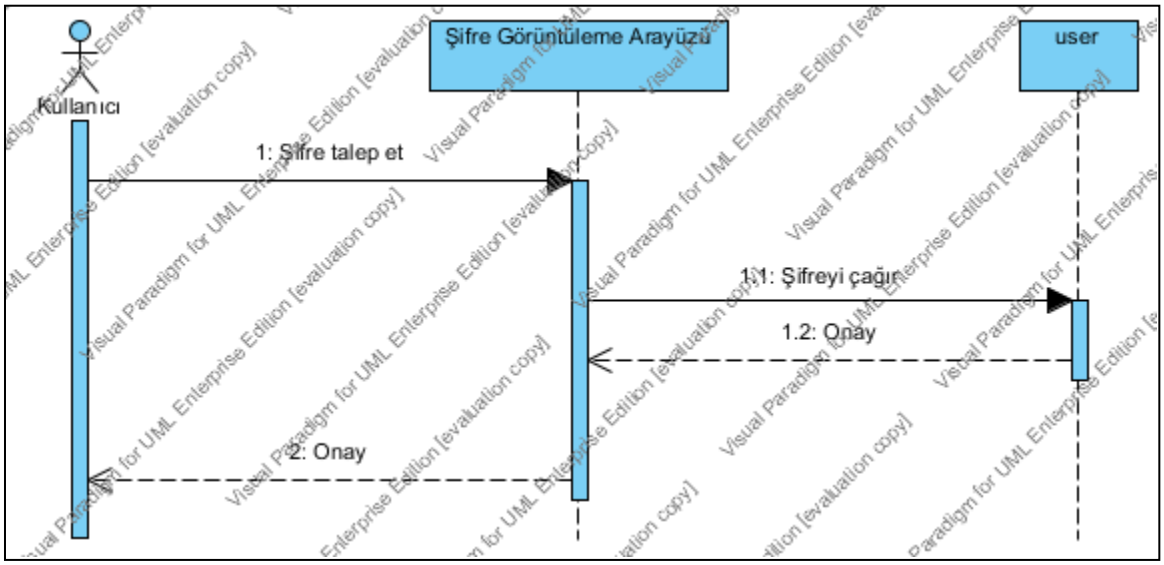
Senaryo 6: Nesne Güncelleme



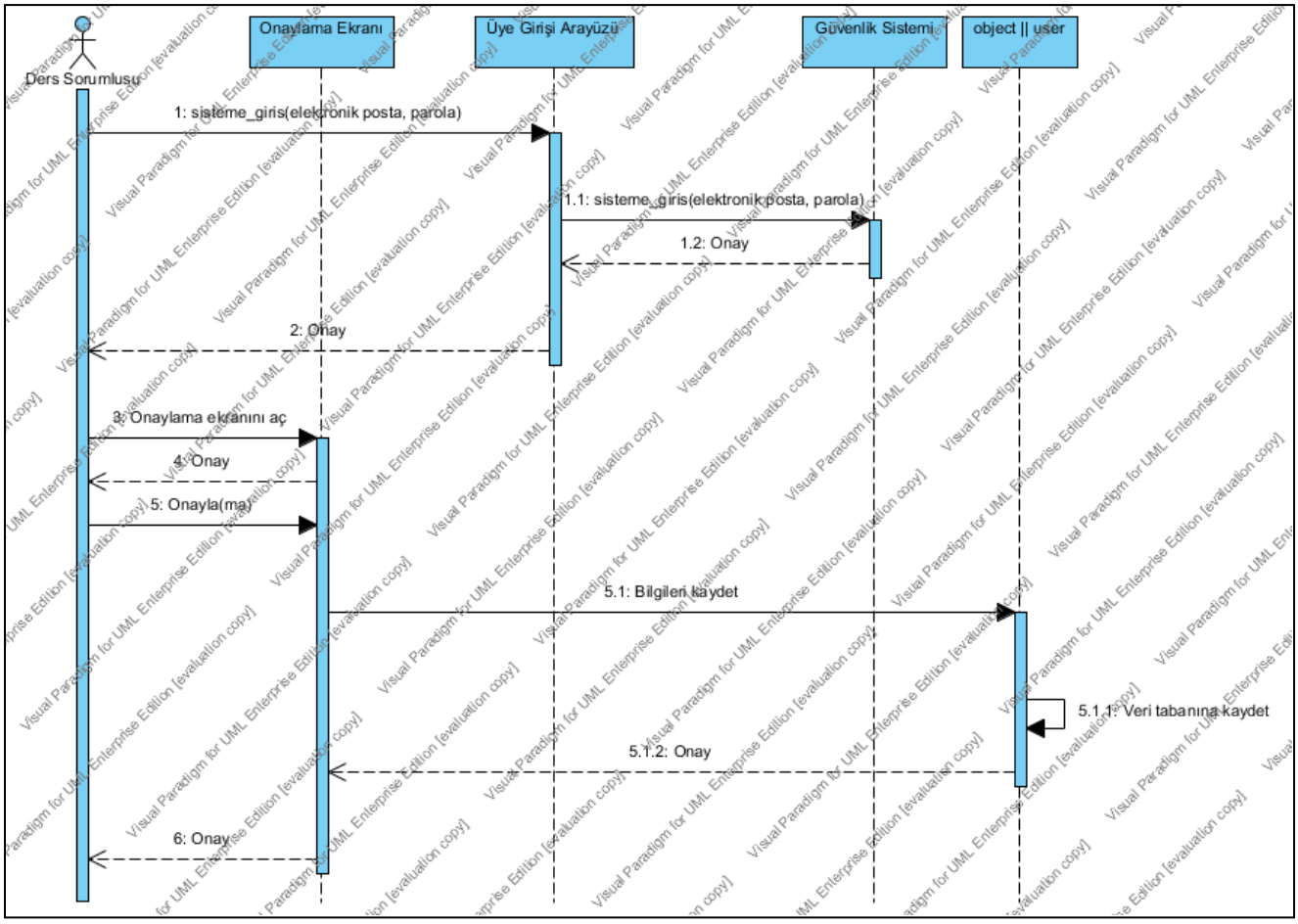
Senaryo 7: Nesne Silme



Senaryo 8: Şifre Gönderme

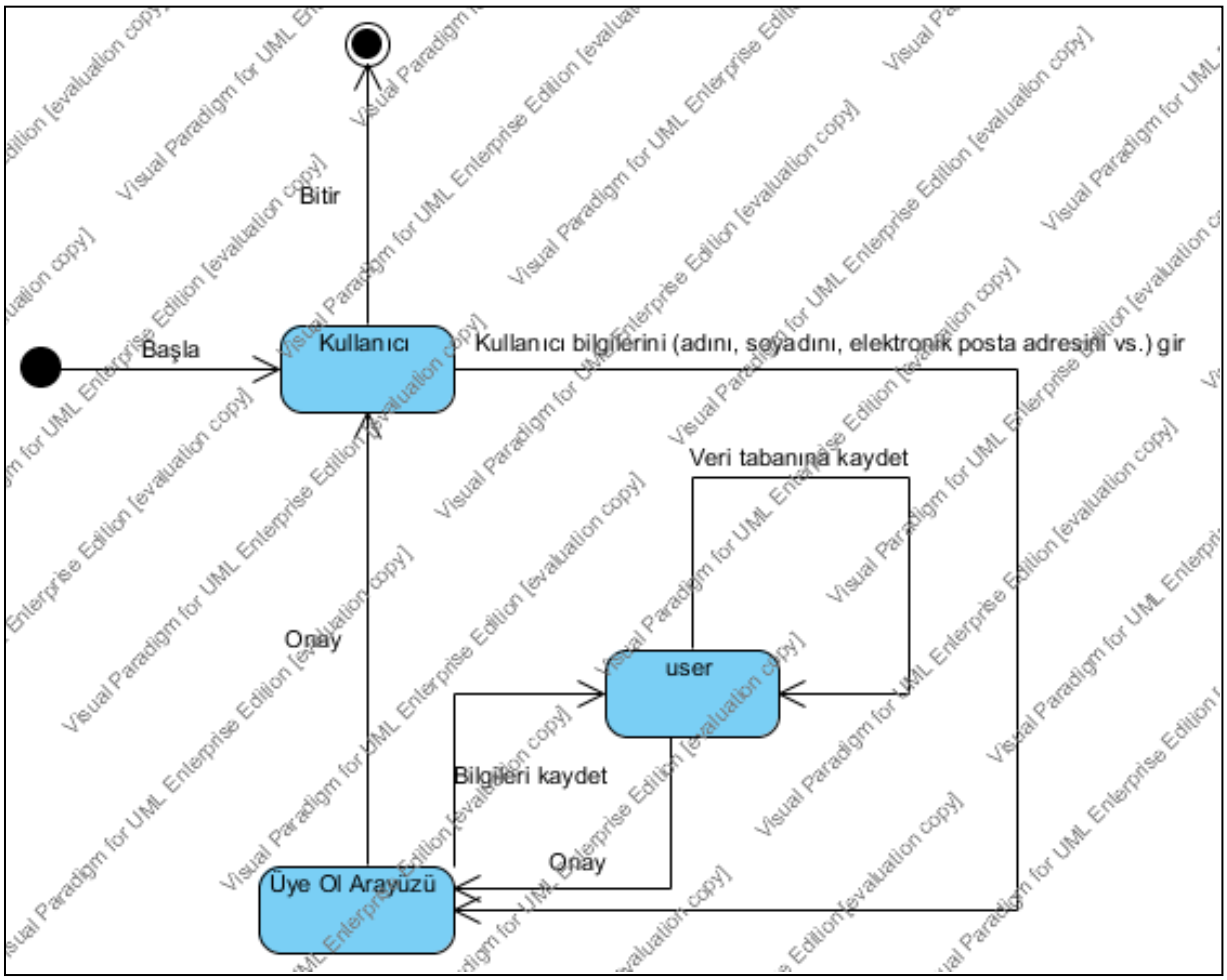


Senaryo 9: Onaylama

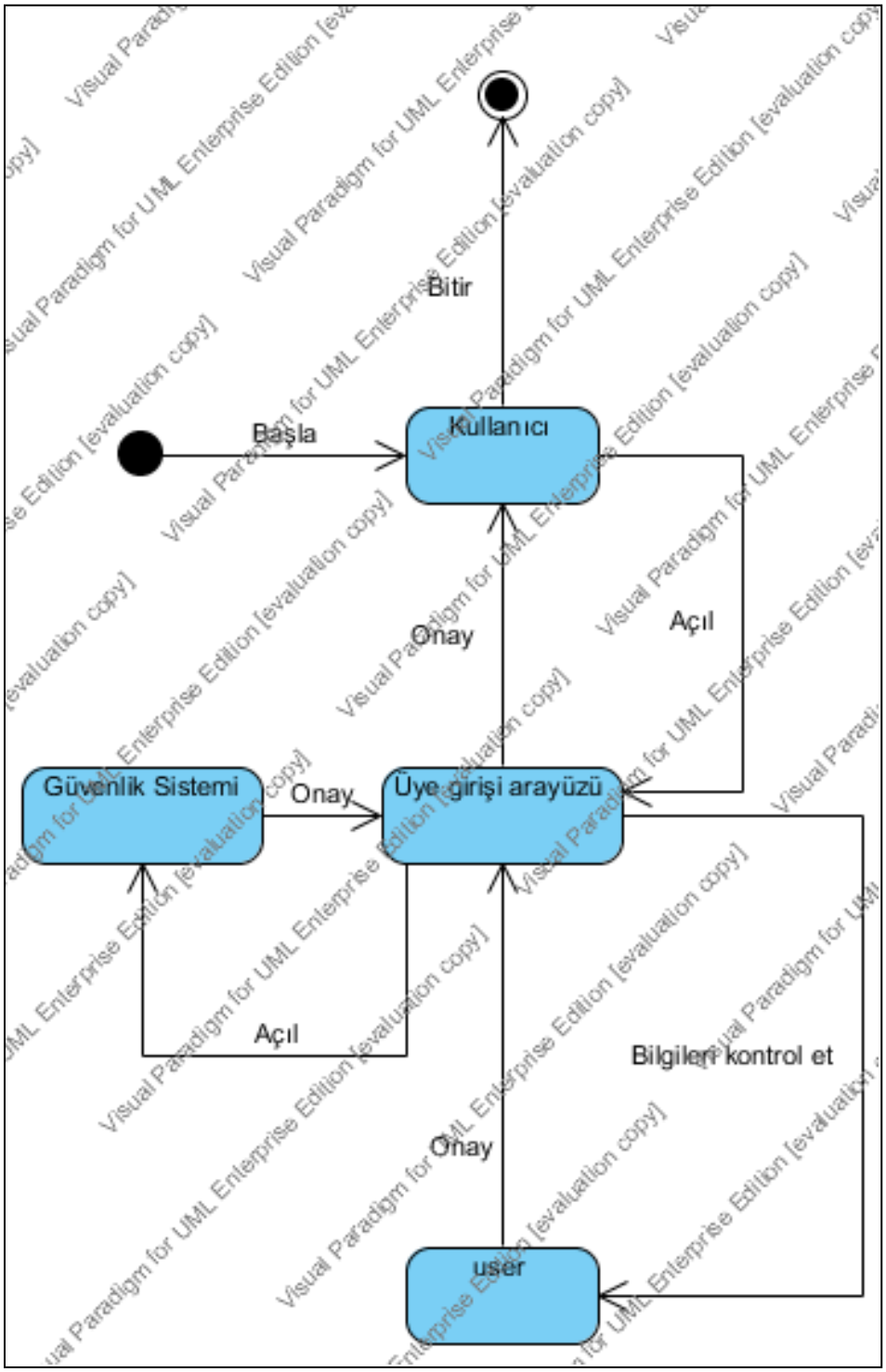


8. Durum Diyagramları

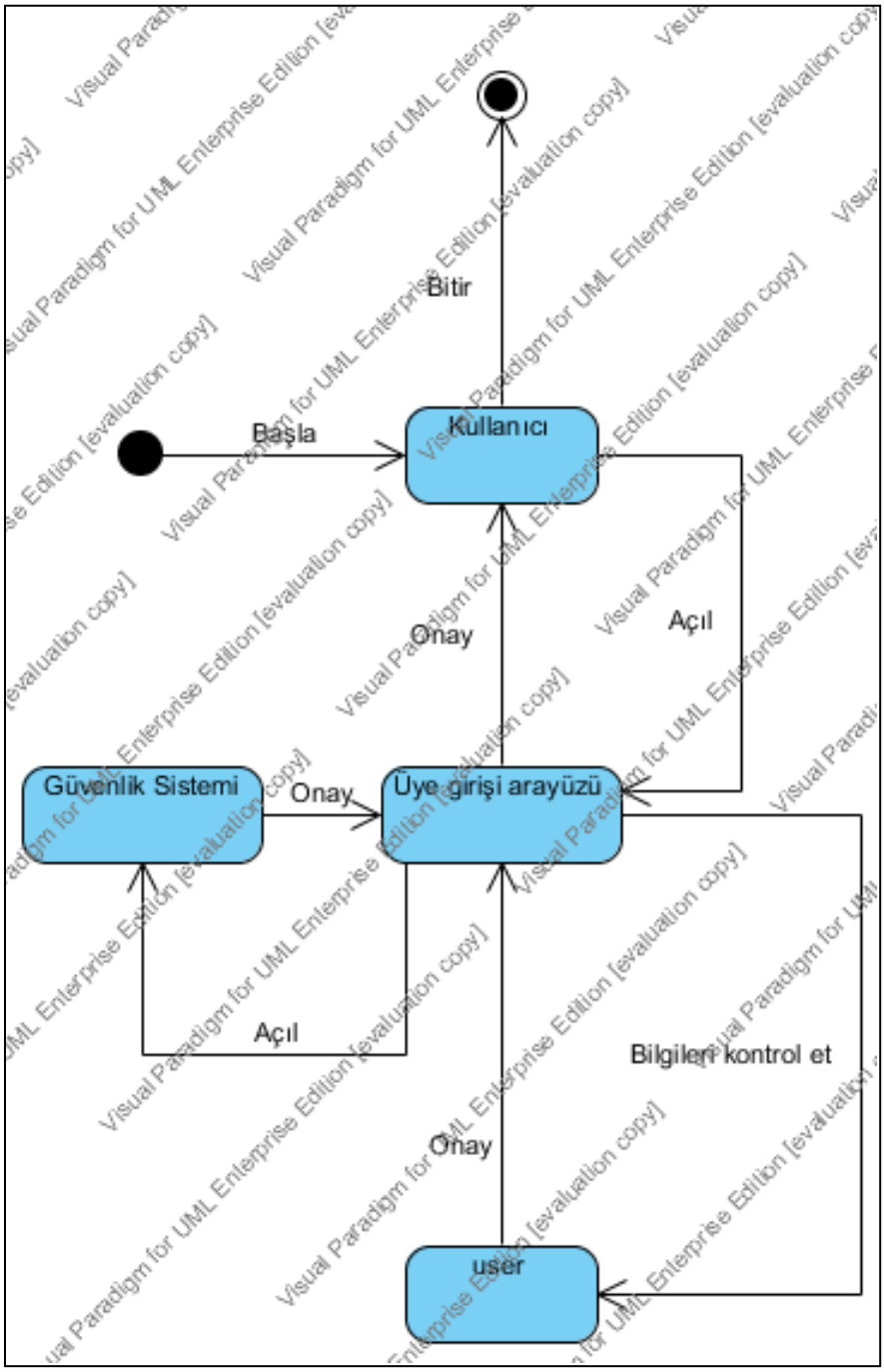
Senaryo 1: Üye Kaydı



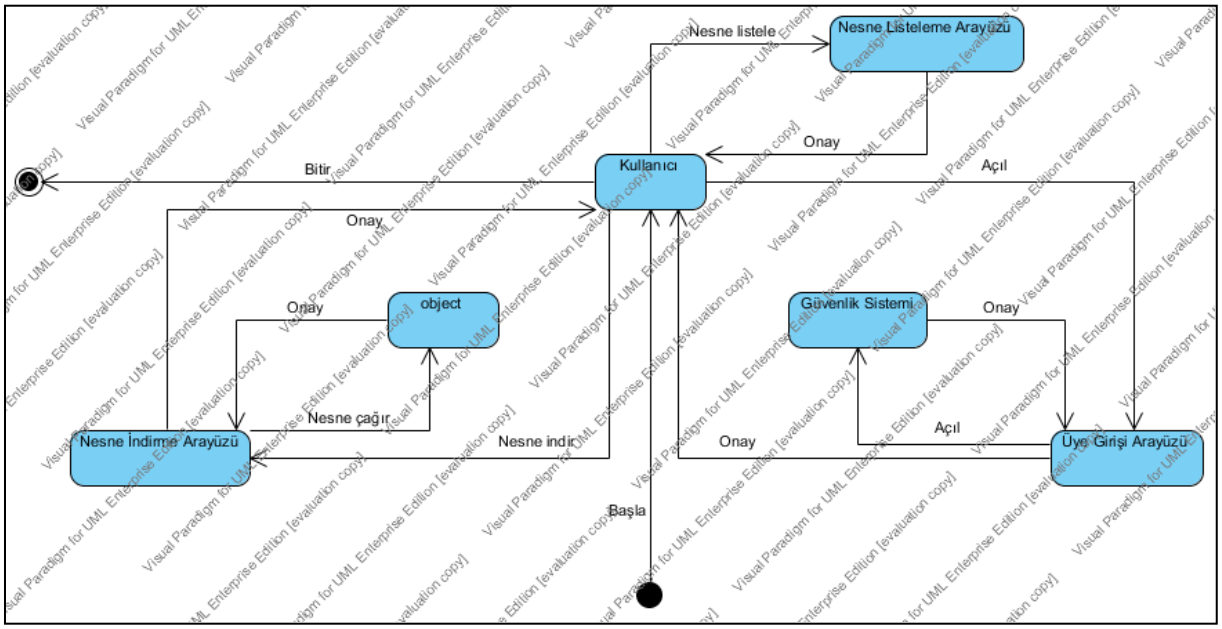
Senaryo 2: Üye Girişi



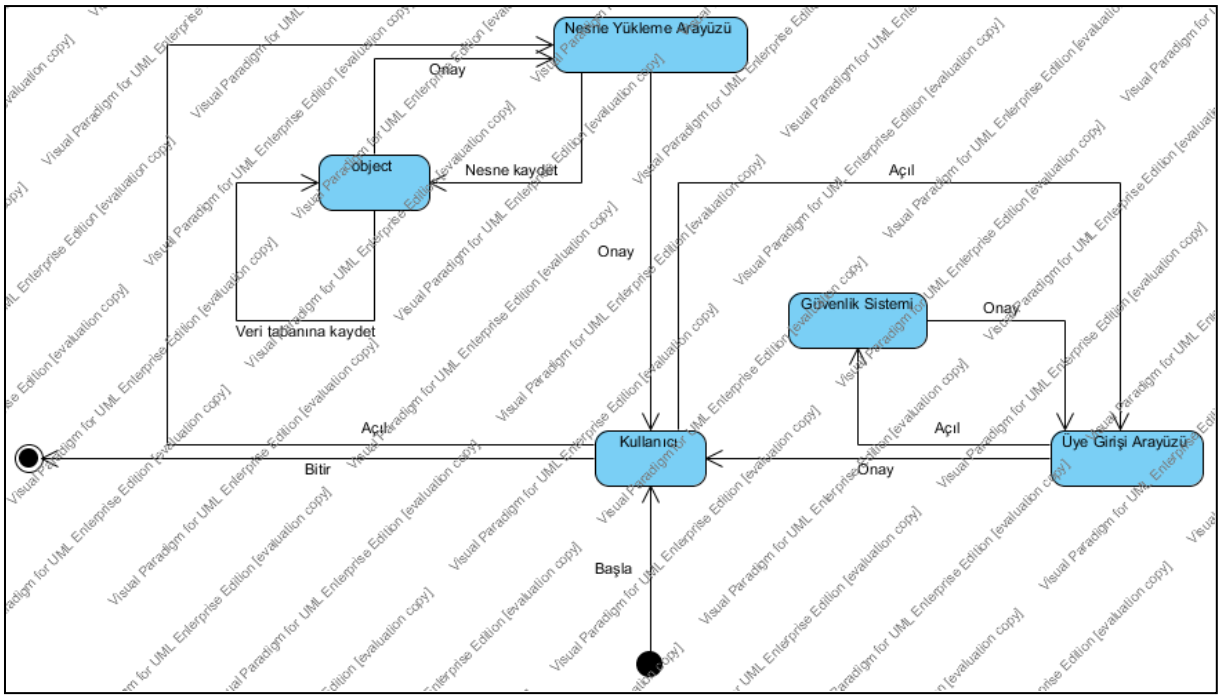
Senaryo 3: Nesne Listeleme



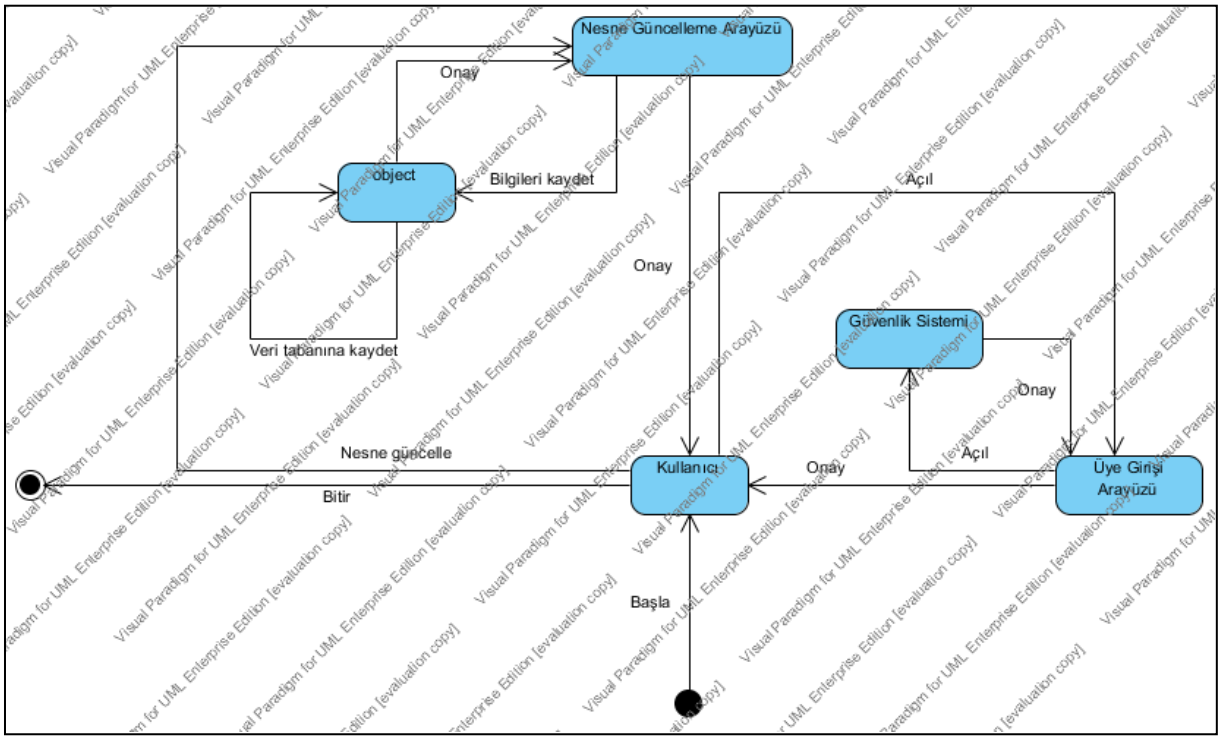
Senaryo 4: Nesne İndirme



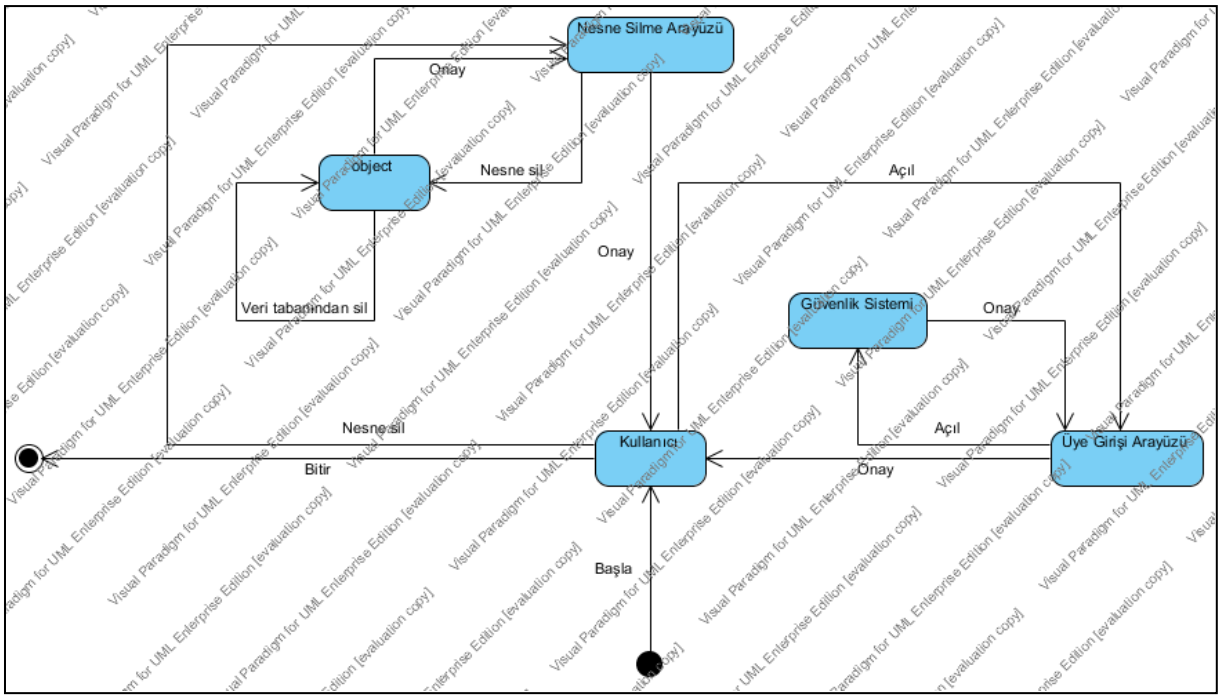
Senaryo 5: Nesne Yükleme



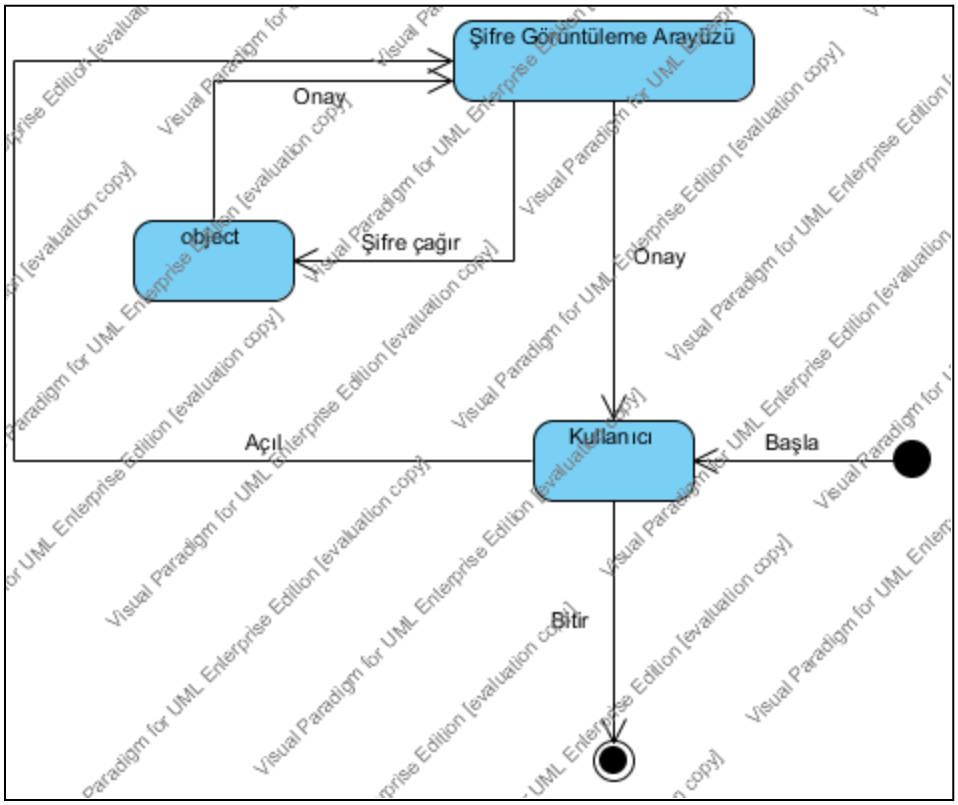
Senaryo 6: Nesne Güncelleme



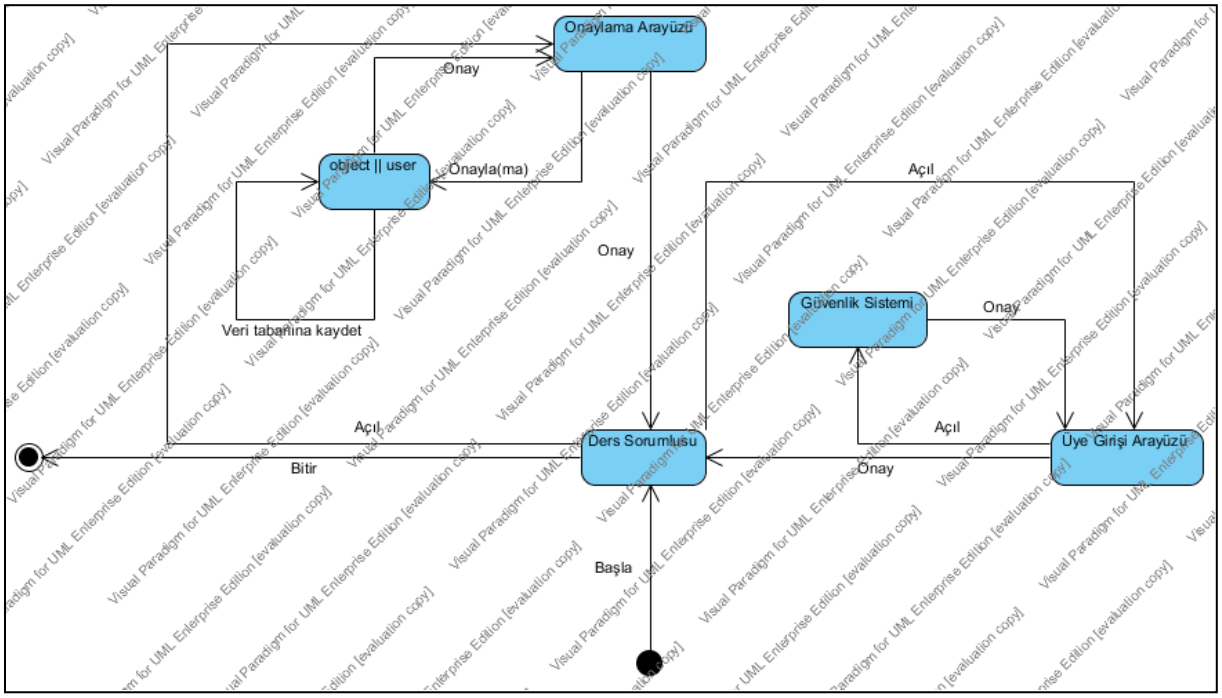
Senaryo 7: Nesne Silme



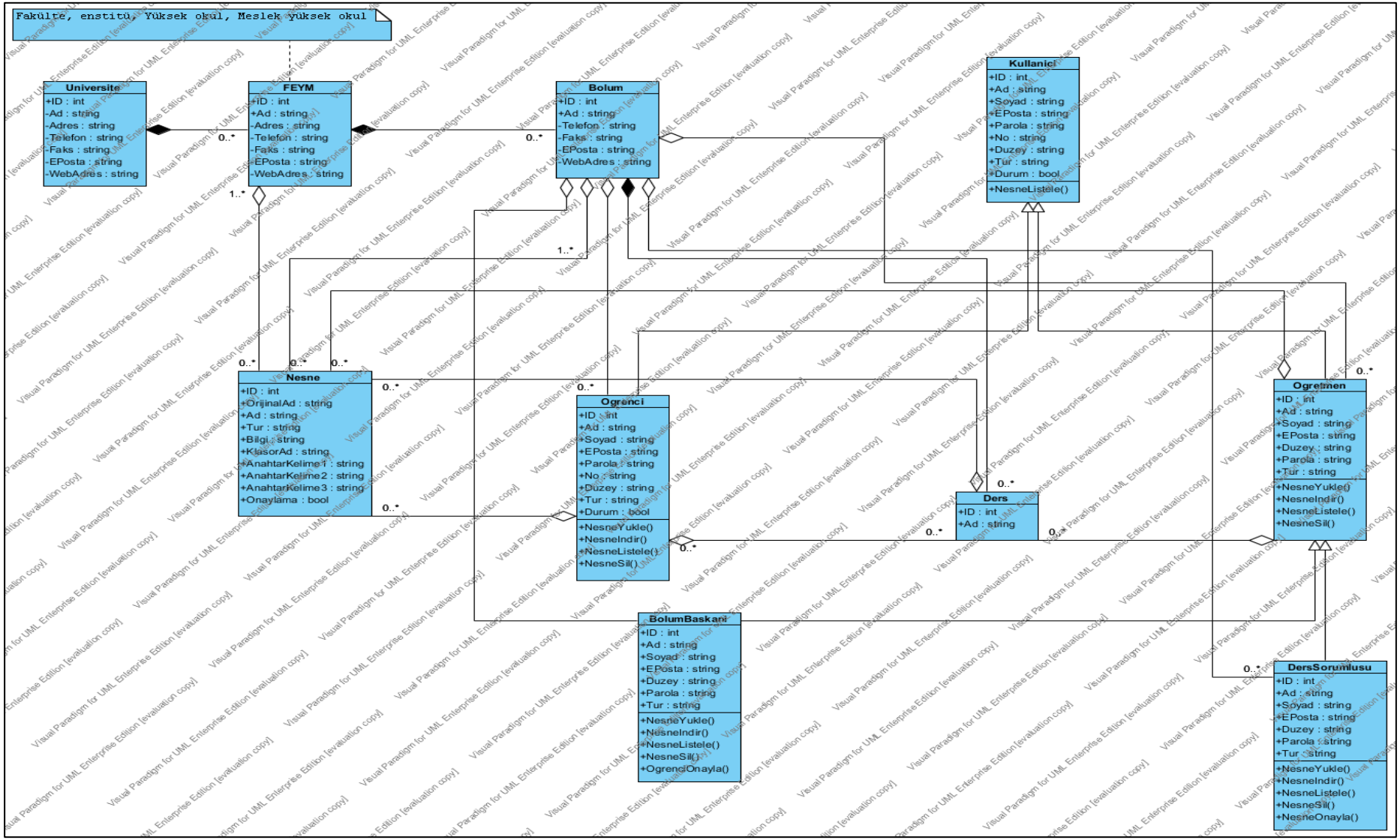
Senaryo 8: Şifre Gönderme



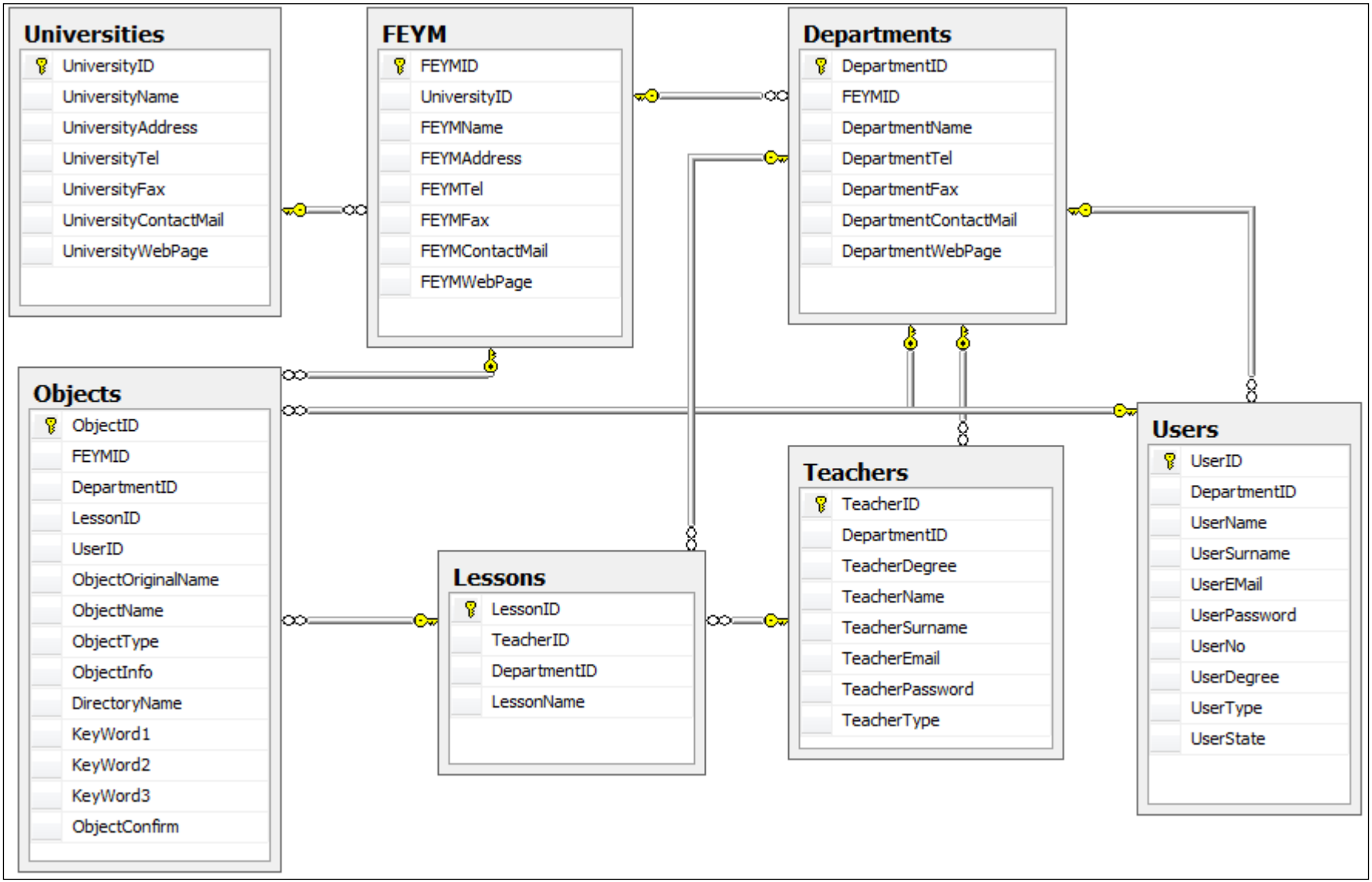
Senaryo 9: Onaylama



9. Nesne İlişkileri Diyagramı



10. Veri Tabanı Modeli



11. Veri Sözlüğü

FEYM	Fakülte / Enstitü / Yüksek Okul / Meslek Yüksek Okul

12. Sorumluluk İlişki Kartları

Kullanıcı Sınıfına Ait Sorumluluk İlişki Kartı

Kullanıcı	
Description: Sistemi kullanan ve sisteme nesne yükleyen kişi	
Responsibilities:	
Name	Collaborator
Nesne yükler.	Nesne
Nesne siler.	Nesne
Nesne günceller	Nesne
Nesne listeler	Nesne
Hesap bilgilerini günceller	Veritabanı
Nesne arar	Nesne
Nesne indirir	Nesne
Sistemi kullanır	Nesne Ambarı
Bilgilerini kaydeder	Veritabanı
Nesne onaylar.	Güvenlik
Öğrenci onaylar.	Güvenlik

Ders Sınıfına Ait Sorumluluk İlişki Kartı

Ders	
Responsibilities:	
Name	Collaborator
Ait olduğu FEYM'i bilir.	FEYM
Ait olduğu bölümü bilir.	Bölüm
Sorumlusu olan öğretmeni	Öğretmen
Sahip olduğu nesneleri bilir.	Nesne

FEYM Sınıfına Ait Sorumluluk İlişki Kartı

FEYM	
Responsibilities:	
Name	Collaborator
Sahip olduğu bölümleri bilir.	Bölüm
Sahip olduğu dersleri bilir.	Ders
Sahip olduğu nesneleri bilir.	Nesne

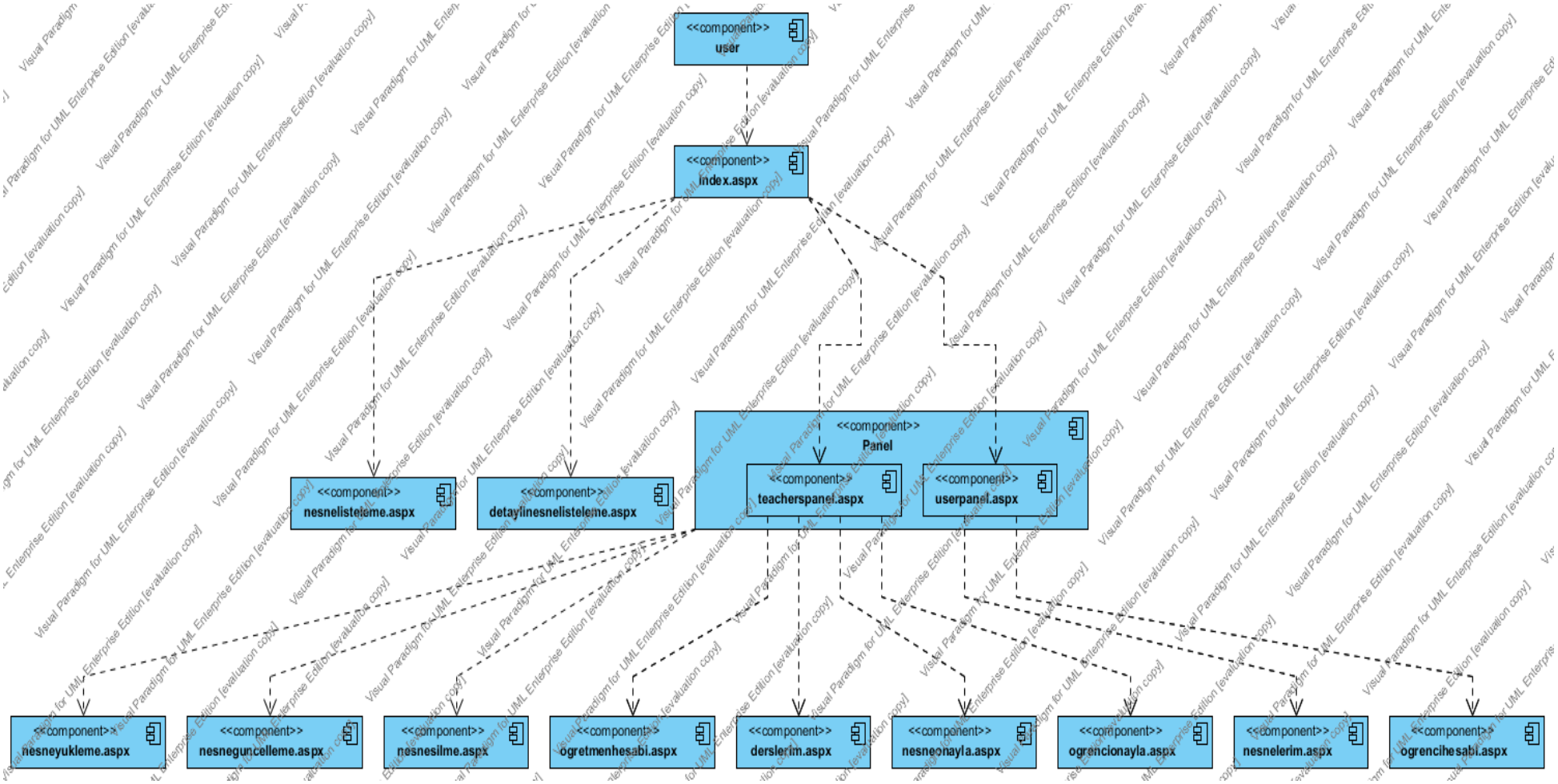
Bölüm Sınıfına Ait Sorumluluk İlişki Kartı

Bölüm	
Responsibilities:	
Name	Collaborator
Sahip olduğu derslerii bilir.	Ders
Sahip olduğu bölüm başkanını bilir.	Bölüm başkanı
Sahip olduğu nesneleri bilir.	Nesne
Sahip olduğu öğretmenleri bilir.	Öğretmen
Sahip olduğu öğrencilerii bilir.	Öğrenci
Ait olduğu FEYM'i bilir.	FEYM

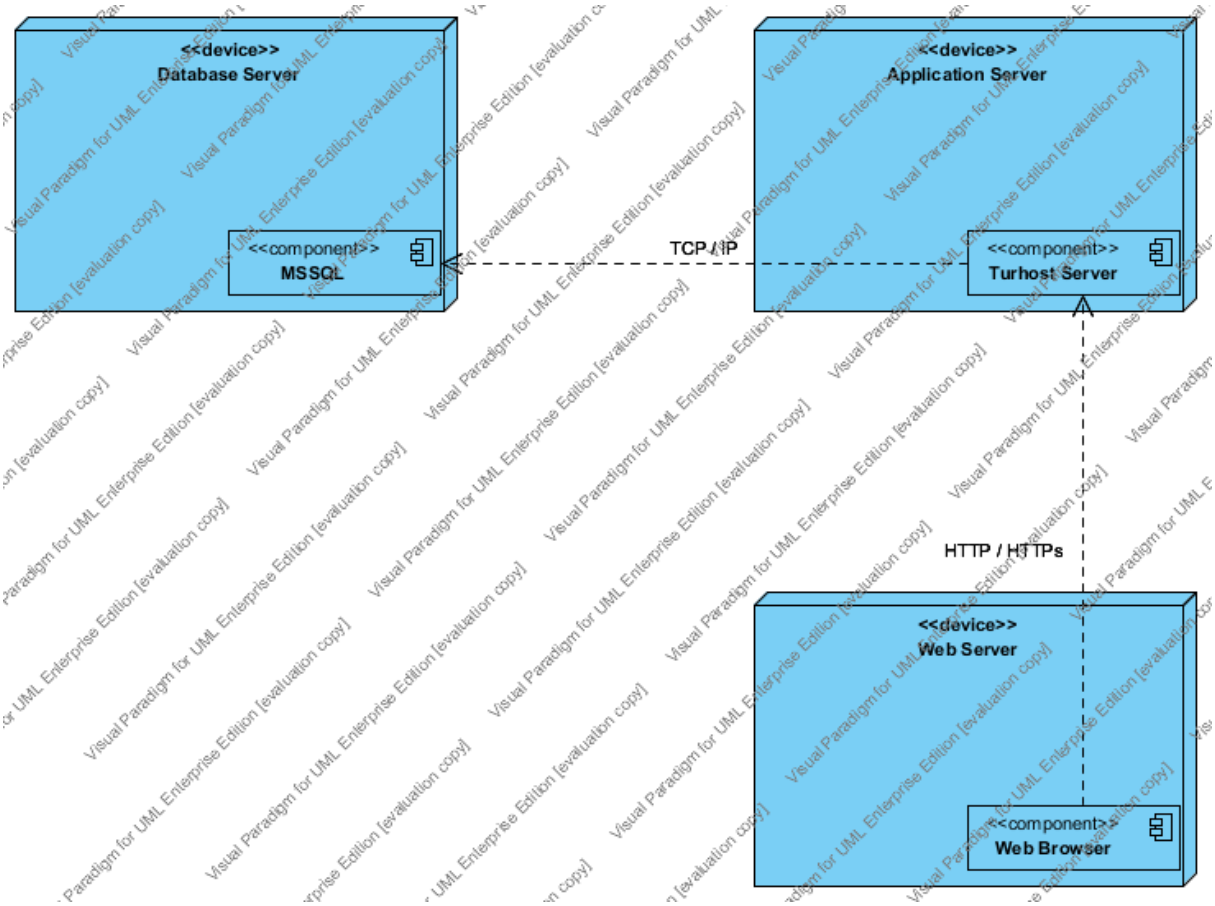
Nesne Sınıfına Ait Sorumluluk İlişki Kartı

Nesne	
Responsibilities:	
Name	Collaborator
Nesnenin sahip olduğu FEYM'i	Fakülte
Nesnenin sahip olduğu bölümü	Bölüm
Nesnenin sahip olduğu dersi bilir.	Ders
Nesnenin kim tarafından yüklendiğini bilir.	Kullanıcı

13. Bileşen Diyagramı



14. Akış Diyagramı



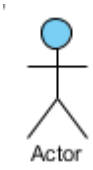
15. Visual Paradigm for UML Enterprise Edition 7.2 Notasyon

Kullanım Durumu (Use Case) Diyagramları

Bu diyagramlar bir sistemin fonksiyonel gereksinimlerini gösterir. Sistemle aktörler arasındaki ilişkiyi tanımlar.

Sistem (System): Dikdörtgen ile gösterilir. Sistemin adı dikdörtgenin üst tarafına yazılır.

Aktör (actor): Sistemdeki aktörleri temsil eder. Aşağıdaki gibi gösterilir. Aktörün adı şeklin aşağısına yazılır.



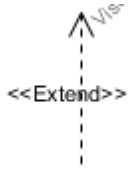
Kullanım durumu (use case): Bir talebe cevap veren sistem davranışıdır. Elips ile gösterilir. Kullanım durumunun ismi elipsin içine yazılır.

İlişki (association): Aktörler ve kullanım durumları arasındaki ilişkidir. Bir çizgi ile gösterilir. İlişki istenirse okun üzerinde isimlendirilebilir.

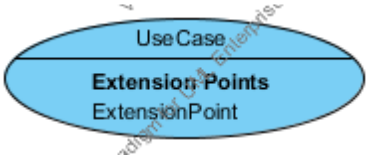
Bağımlılık (dependency): İki kullanım durumu arasında gerçekleşir. Bir kullanım durumunun diğer kullanım durumuna bağımlı olduğunu belirtir. Kesikli bir ok ile gösterilir. Okun gösterdiği kullanım durumu, bağımlı olunan kullanım durumu, diğer kullanım durumu ise bağımlı olan kullanım durumudur. Bu gösterimde dilenirse, bağımlılık durumu okun alt tarafına yazılabilir.

Dahil etme (include): Bir bağımlılık durumudur. Bağımlı olunan kullanım durumunun, bağımlı olan kullanım durumuna dahil olduğunu belirtir. Yani bağımlı olan kullanım durumunun gerçekleşmesi için bağımlı olunan kullanım durumu gerçekleşmek zorundadır. Aşağıdaki gibi gösterilir. Şeklin altına istenirse açıklama yazılabilir.

Uzatma (extend): Bir bağımlılık durumudur. Bağımlı olunan kullanım durumu, bağımlı olan kullanım durumuna uzatılabilir. Yani bağımlı olunan kullanım durumu gerçekleşirken, bağımlı olan kullanım durumunun gerçekleşme ihitimali vardır. Aşağıdaki gibi gösterilir. Şeklin altına istenirse açıklama yazılabilir.



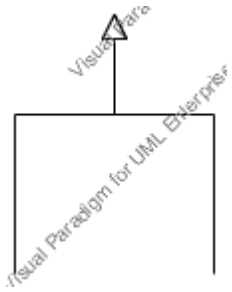
Bu durum kullanıldığında bağımlı olunan kullanım durumunda uzamaların nerede olacağı istenirse yazılabilir. Bu yazım, aşağıdaki şekildeki gibi "Extension Points" başlığı altında yapılır.



Genelleştirme (generalization): Kullanım durumlarının alt kullanım durumları olduğunu beilrtir. Aşağıdaki gibi bir ok ile gösterilir. Okun gösterdiği kullanım durumu, üst kullanım durumu,

diğeri ise alt kullanım durumudur. Okun üzerine istenirse açıklama yazılabilir. Aktörler için de kullanılabilir.

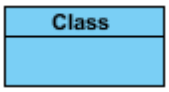
Birden fazla alt kullanım durumu olduğunda, bunlar aşağıdaki gibi hiyerarşik bir biçimde de gösterilebilir.



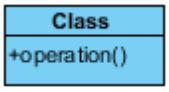
Sınıf (Class) Diyagramları

Sınıf diyagramları, sınıfları ve onların birbirleriyle olan ilişkilerini resmeder.

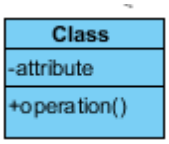
Sınıf (class): Sınıfları temsil eder. Sınıflar aşağıdaki gibi bir dikdörtgen ile gösterilir. Dikdörtgenin üst kısmına sınıfın ismi yazılır. Sınıf eğer soyutsa sınıf ismi italik yazılır.



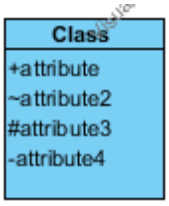
Sınıf gösteriminde daha fazla ayrıntıya girilebilir. Sınıfın ismi ve o sınıfın yaptığı operasyonlar aşağıdaki gibi gösterilir. Eğer operasyon soyutsa italik olarak yazılır.



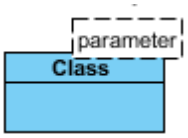
Sınıf gösteriminde, sınıfın özellikleri de aşağıdaki gibi eklenebilir.



Sınıf gösteriminde, sınıfın özelliklerinin ve operasyonlarının görünülürlüğü de belirtilebilir. “public”, “protected”, “package” ve “private” görünülürlük için sırasıyla “+”, “~”, “#” ve “-” işaretleri özelliklerin önüne yazılır.



Eğer sınıf bir şablonsa aşağıdaki gibi, parametre sınıfın üst tarafındaki dikdörtgene yazılır.



İlişki (association): Sınıflar arasındaki ilişkidir. Bir çizgi ile gösterilir. Sınıflar arsında bir sahip olma ilişkisi belirtir. İlişki istenirse okun üzerinde isimlendirilebilir. Bu ilişkide, istenirse bir sınıf türünden nesnelerin sayısı o sınıfın hemen yanına yazılabilir (multiplicity). Bu yazımda “*” birden fazla anlamına gelir. “a..b”, nesnelerin sayısının, a sayısından b sayısına kadar olan sayılardan (a ile b dahil) biri olduğu anlamına, “a,b” ise, nesnelerin sayısının, a ya da b kadar olduğu anlamına gelir. Eğer, nesne sayısı 1 ise, sınıfın yanına istenirse nesne sayısı yazılmayabilir.

İlişki sınıfı (association class): Sınıflar arası bir ilişkiden doğan sınıftır. Kesikli bir çizgi ile gösterilir. Çizgi ilişki ile doğan sınıf arasındadır.

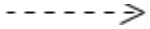
Kümeleme (aggregation): Bir sınıfın başka bir sınıfın nesnelerinden oluştuğunu gösterir. Aşağıdaki gibi gösterilir. Oluşan sınıf dörtgenin olduğu taraftadır. Bileşimden (composition) farkı, dörtgenin olmadığı tarafta olan sınıfın nesnelerinin, başka sınıfın parçaları da olabilmesidir. İstenirse çizginin aşağısına açıklama yazılabilir. Bu ilişkide de istenirse nesnelerin sayısı yanına yazılabilir.

Bileşim (composition): Bir sınıfın başka bir sınıfın nesnelerinden oluştuğunu gösterir. Aşağıdaki gibi gösterilir. Oluşan sınıf dörtgenin olduğu taraftadır. Kümelemeden (aggregation) farkı, dörtgenin olmadığı tarafta olan sınıfın nesnelerinin, başka sınıfın parçaları olamamasıdır. İstenirse çizginin aşağısına açıklama yazılabilir. Bu ilişkide de istenirse nesnelerin sayısı yanına yazılabilir.

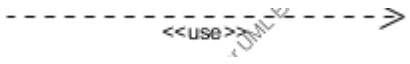
Genelleştirme (generalization): Kalıtım. Aşağıdaki gibi gösterilir. Alt sınıflar üçgenin olmadığı taraftadır. İstenirse okun üzerine açıklama yazılabilir.

Gerçekleştirme (realization): Bir sınıfın, bir arayüzün (interface) özelliklerini ve operasyonlarını uygulayabileceğini belirtir. Kesikli bir ok ile gösterilir. Okun gösterdiği tarafta arayüz bulunur. İstenirse okun üzerine açıklama yazılabilir.

Bağımlılık (dependency): Bir sınıf bir sınıfı kullandığında gösterilir. Kesikli ok ile gösterilir. Okun gösterdiği taraftaki sınıf, kullanılan sınıftır. İstenirse okun altına açıklama yazılabilir.

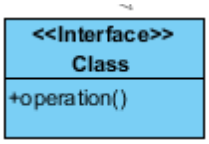


Kullanım (usage): Bağımlılığın genel halidir. Aşağıdaki gibi gösterilir. İstenirse okun altına açıklama yazılabilir.



Arayüz (interface): Bir sınıf kalıbıdır. Bir veya daha fazla sınıfa veya bileşene hizmet etme amaçlı bir takım operasyonlardır. Aşağıdaki gibi arayüzün ismi "<<Interface>>" yazısının altına yazılır.

Arayüz gösteriminde operasyonlar da yazılabilir.



İşbirliği (collaboration): Bir takım elemanların bir amaç doğrultusunda işbirliği ile çalışmasıdır. Aşağıdaki gibi gösterilir. İşbirliğinin ismi elipsin içine yazılır.

Not (note): Açıklama notlarıdır. Aşağıdaki gibi bir şeklin içine yazılır.

Bağlama (anchor): Bir notu ilgili oluğu nesneye bağlamak için kullanılır. Kesikli çizgi ile gösterilir. İstenirse çizginin üzerine açıklama yazılabilir.

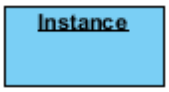
Kısıtlama (constraint): Bir notu ilgili oluğu nesneye bağlamak için kullanılır. Eğer not bir kısıtlama içeriyorsa kullanılır. Kesikli çizgi ile gösterilir. Bağlamadan farkı çizgilerin biraz daha uzun olmasıdır. İstenirse çizginin üzerine açıklama yazılabilir.

İçerme (containment): Bir nesnenin bir diğer nesneyi içerdiğini göstermek için kullanılır. Aşağıdaki gibi gösterilir. Dairenin olduğu taraf içeren nesnedir.

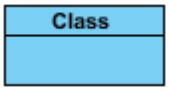
Nesne (Object) Diyagramları

Nesnelerin birbirleriyle olan ilişkilerini gösterir.

Örnek belirleme (instance specification): Nesnelere bu isim verilir. Aşağıdaki gibi gösterilir. Nesne isimleri dikdörtgenin içinde altı çizili olarak yazılır.

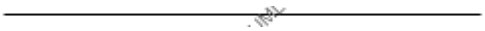


Sınıf (class): Sınıf diyagramlarındaki sınıflarla aynıdır. Aşağıdaki gibi gösterilir.



Bağ (link): İki nesne arasında ilişki olduğunu gösterir. Bir çizgi ile gösterilir. İstenirse çizginin altına açıklama yazılabilir.

İlişki (association): Bir nesne veya sınıf ile bir sınıf arasındaki ilişkidir. Bir çizgi ile gösterilir. Sınıflar arsında bir sahip olma ilişkisi belirtir. İlişki istenirse okun üzerinde isimlendirilebilir. Bu ilişkide, istenirse nesnelerin veya bir sınıf türünden nesnelerin sayısı o sınıfın hemen yanına yazılabilir (multiplicity). Bu yazımda "*" birden fazla anlamına gelir. "a..b", nesnelerin sayısının, a sayısından b sayısına kadar olan sayılardan (a ile b dahil) biri olduğu anlamına, "a,b" ise, nesnelerin sayısının, a ya da b kadar olduğu anlamına gelir. Eğer, nesne sayısı 1 ise, sınıfın ya da nesnenin yanına istenirse nesne sayısı yazılmayabilir.

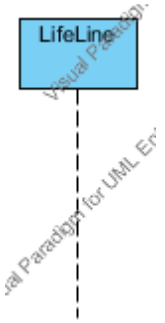


Bileşim (composition): Bir nesnenin başka nesnelerden oluştuğunu gösterir. Aşağıdaki gibi gösterilir. Oluşan nesne dörtgenin olduğu taraftadır. Kümelemeden (aggregation) farkı, dörtgenin olmadığı tarafta olan nesnelerin, başka nesnelerin parçaları olamamasıdır. İstenirse çizginin aşağısına açıklama yazılabilir. Bu ilişkide de istenirse nesnelerin sayısı yanına yazılabilir.

Ardışıklık (Sequence) Diyagramları

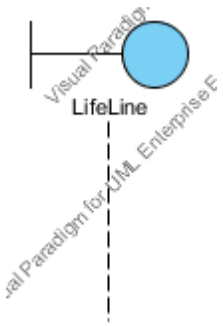
Süreçlerin birbirleriyle olan etkileşiminin nasıl ve hangi sırada olduğunu gösterir.

Yaşam çizgisi (lifeline): Ardışıklık diyagramındaki nesnelerdir. Aşağıdaki gibi gösterilir. Dikdörtgenin içine nesne adı yazılır.



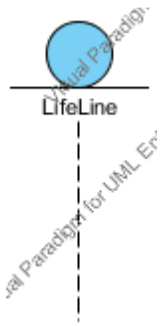
Aktör (actor): Bir nesne türüdür. Aşağıdaki gibi aktörün adı dikdörtgenin üstüne yazılır. Dikdörtgen, aktörün hep aktif olduğunu gösterir.

Sınır yaşam çizgisi (boundary lifeline): Kullanıcı arayüz ekranı ya da bir giriş/çıkış aygıtını gösterir. Aşağıdaki gibi kesikli çizginin üzerine sınır adı yazılır.

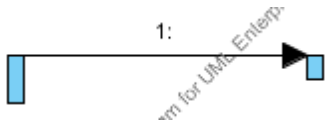


Kontrol yaşam çizgisi (control lifeline): Yapılmış bir işin ve o işin nasıl ve ne zaman yapıldığının kontrolünü yapar. Aşağıdaki gibi kesikli çizginin üzerine kontrol adı yazılır.

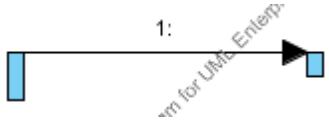
Varlık yaşam çizgisi (entity lifeline): Dosya gibi içinde veri tutulan sürekli bir nesnedir. Genel olarak veri tabanında bir tablo tarafından uygulanır. Aşağıdaki gibi kesikli çizginin üzerine varlık adı yazılır.



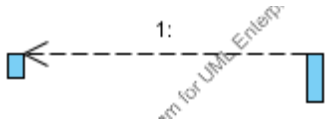
Mesaj (message): Nesnelerin birbirlerine gönderdiği mesajlar aşağıdaki gibi bir ok ile gösterilir. Bu şekilde küçük dikdörtgenler aktivasyon kalıplarıdır. Aktivasyon kalıpları, nesnenin yapmakla yükümlü olduğu operasyonların yürütülüşünü gösterir. Başka bir deyişle nesnenin ne zaman aktif olduğunu gösterir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.



Çağrı mesajı (call message): Senkronize mesajlardır. Mesajı gönderen nesnenin işleyişine devam etmesi için, mesajı alan nesne bir geri dönüş mesajı göndermelidir. Aşağıdaki gibi gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.

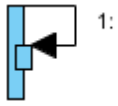


Geri dönüş mesajı (return message): Geri dönüş mesajlarını belirtir. Aşağıdaki gibi kesikli bir okla gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.

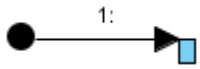


Kendi kendine mesaj (self message): Bir nesnenin kendi kendine mesaj göndermesidir. Aşağıdaki gibi bir okla gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.

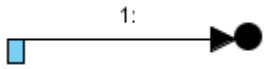
Tekrarlayan mesaj (recursive message): Tekrarlayan mesajlarda aktivasyon kalıbının üzerinde bir başka aktivasyon kalıbı bulunur. Aşağıdaki gibi bir okla gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.



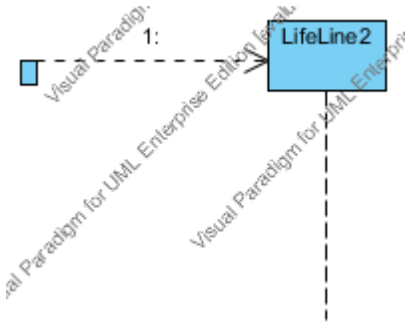
Bulunmuş mesaj (found message): Bilinmeyen bir yerden veya başka bir diyagramdan, gelen mesajlardır. Aşağıdaki gibi bir daireden bir aktivasyon kalıbına çizilen bir ok ile gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.



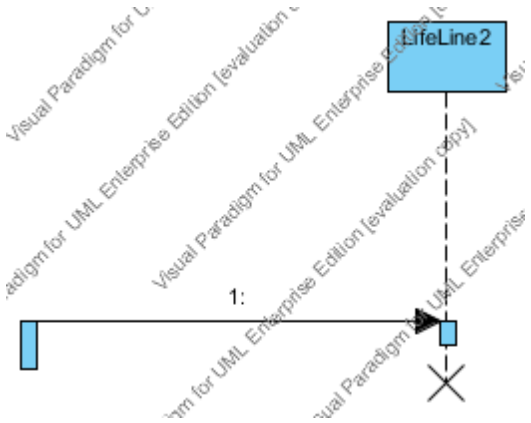
Kayıp mesaj (lost message): Başka bir diyagramdaki nesneye giden mesajlardır. Aşağıdaki gibi bir aktivasyon kalıbından bir daireye çizilen bir ok ile gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.



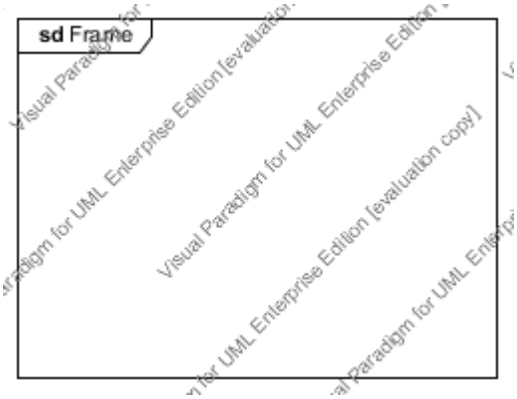
Yaratma mesajı (create message): Bir nesnenin gönderdiği mesajla bir nesnenin yaşam çizgisini yaratmasıdır. Aşağıdaki gibi bir aktivasyon kalıbından bir nesneye çizilen kesikli bir ok ile gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir.



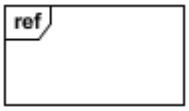
Yok etme mesajı (destroy message): Bir nesnenin yaşam çizgisini yok eder. Aşağıdaki gibi bir aktivasyon kalıbından bir nesnenin yaşam çizgisine çizilen kesikli bir ok ile gösterilir. Okun üzerine aşağıdaki gibi bir mesaj numarası yazılması zorunludur. İstenirse mesajın ne olduğu yazılabilir. Yaşam çizgisinin sonlanması aşağıdaki gibi bir çarpı ile gösterilir.



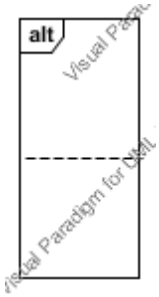
Çerçeve (frame): Ardışıklık diyagramının aşağıdaki dikdörtgen içinde kalan bütününe ya da bir kısmına isim vermekte kullanılır. Aşağıdaki gibi gösterilir. Sol üst köşeye, “sd” kısaltmasının sağına ilgili kısma verilen isim yazılır. İsimlendirmede özel kısaltmalar kullanılabilir. “opt” kısaltması işleyişin gerçekleşmesinin bir koşula bağlı olduğunu gösterir. Bu koşul “opt” kısaltmasının sağına “[]” parantezleri arasına yazılabilir. “neg” kısaltması hatalı bir etkileşim olduğunu, “region” ise sadece bir thread işleyebileceğini gösterir.



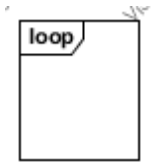
Etkileşim kullanımı (interaction use): Ardışıklık diyagramının aşağıdaki dikdörtgen içinde kalan bütününde ya da bir kısmında, başka bir diyagramda tanımlanan bir etkileşime referans verir.



Alt. birleşik parçası (alt. combined fragment): Ardışıklık diyagramının aşağıdaki dikdörtgen içinde kalan bütününde ya da bir kısmında, her işleyiş için bir koşul yazılır. En üst dikdörtgende sol üst köşeye, "alt" kısaltmasının sağına ilgili kısma verilen koşul yazılır. Bu dikdörtgenin aşağısında kalan dikdörtgen ya da dikdörtgenlerin sol üst köşesine, ilgili kısma verilen koşul yazılır. Bu birleşik parça aynı anda birden fazla mesaj gönderildiğini belirtmek için de kullanılır. Bu, her bir dikdörtgende bir mesaj olmasına dikkat edilerek ve alt kısaltmasının sağına "[par]" kısaltması yazılarak belirtilir.



Döngü birleşik parçası (loop combined fragment): Ardışıklık diyagramının aşağıdaki dikdörtgen içinde kalan bütününde ya da bir kısmında, döngü olması durumunda kullanılır.

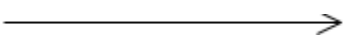


Durum (State Machine) Diyagramları

Sistemin davranışlarını gösterir.

Durum (state): Bir nesnenin o anki durumunu belirtir. Aşağıdaki gibi gösterilir. Durumun ismi şeklin içine yazılır.

Geçiş (transition): Bir durumdan bir başka duruma geçildiğinde kullanılır. Aşağıdaki gibi gösterilir. Okun altına açıklama yazılabilir.



İlk sahte durum (initial pseudo state): Diyagramın başında geçişler sırasında neler olduğunu belirtmek için kullanılır. Aşağıdaki gibi gösterilir.

Son durum (final state): İşlemlerin bittiğini gösterir. Aşağıdaki gibi gösterilir.

Sonlandırıcı (terminate): Bir sahte durumdur. Bir makinenin sonlanması durumunu belirtir. Aşağıdaki gibi gösterilir.

Kavşak (junction): Bir ya da daha fazla durumdan, bir ya da daha fazla durum oluşturmak gerektiğinde kullanılır. Aşağıdaki gibi gösterilir.

Tercih (choice): İki farklı durumdan sadece birinin seçilmesi durumunda kullanılır. Aşağıdaki gibi gösterilir. İstenirse dörtgenin içine açıklama yazılabilir.

Çatal (fork): Bir durumdan iki durum oluşturmak gerektiğinde kullanılır. Aşağıdaki gibi gösterilir.

Birleşim (join): İki durumdan bir durum oluşturmak gerektiğinde kullanılır. Aşağıdaki gibi gösterilir.

Giriş noktası (entry point): Opsiyonel bir başlangıç noktası olduğunu belirtmek için kullanılır. Aşağıdaki gibi gösterilir. İstenirse dairenin altına başlangıç koşulu yazılabilir.

Çıkış noktası (exit point): Olası bir sonlanma durumudur. Aşağıdaki gibi gösterilir. İstenirse dairenin altına açıklama yazılabilir.

Yüzeysel tarih (shallow history): Bir durumu, daha önce gerçekleşen bir durumun başına getirmek için kullanılır. Aşağıdaki gibi gösterilir.

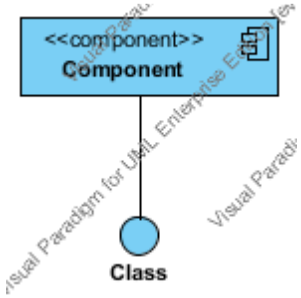
Derin tarih (deep history): Bir durumu, daha önce gerçekleşen bir duruma getirmek için kullanılır. Shallow history, bir durumu daha önce gerçekleşen bir durumun başına getirmek için kullanılırken, deep history daha önce gerçekleşen bir durumun kaldığı yerden devam ettiğini belirtir. Aşağıdaki gibi gösterilir.

Bileşen (Component) Diyagramları

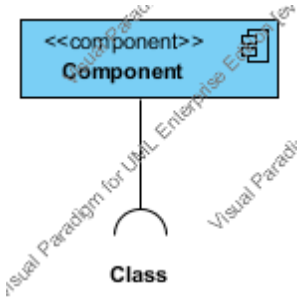
Sistemin fiziksel görünümünü sağlar.

Bileşen (component): Sistemin yapıtaşlarıdır. Yazılımın yeniden kullanılabilen ve değiştirilebilen parçalarıdır. Aşağıdaki gibi gösterilir. "<<component>>" yazısının altına bileşenin ismi yazılır.

Bir bileşen bir arayüzü (interface) sağlayabilir. Bu aşağıdaki gibi gösterilir. Arayüzün ismi dairenin altına yazılır.

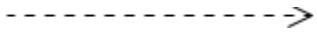


Bir bileşenin gerçeklenmesi için bir arayüz gerekiyor olabilir. Bu aşağıdaki gibi gösterilir. Arayüzün ismi yarım çemberin altına yazılır.



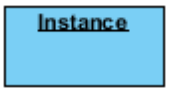
Arayüz (interface): Birbirleriyle mantıksal olarak ilgili olan operasyonların toplandığı sınıftır. Aşağıdaki gibi gösterilir. Arayüzün altına sınıfın ismi yazılır.

Bağımlılık (dependency): İki arayüzün birbirine bağımlı olduğunu belirtir. Kesikli bir ok ile gösterilir. Okun gösterdiği bileşen, bağımlı olunan bileşen, diğer bileşen ise bağımlı olan bileşendir. Kesikli bir ok ile gösterilir. Bu gösterimde dilenirse, bağımlılık durumu okun alt tarafına yazılabilir.



Gerçekleştirme (realization): Bir bileşenin, bir arayüzün (interface) özelliklerini ve operasyonlarını uygulayabileceğini belirtir. Düz bir çizgi ile gösterilir. İstenirse çizginin altına açıklama yazılabilir.

Örnek belirleme (instance specification): Nesnelere bu isim verilir. Aşağıdaki gibi gösterilir. Nesne isimleri dikdörtgenin içinde altı çizili olarak yazılır.



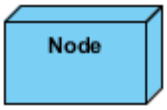
Bağ (link): İki nesne arasında ilişki olduğunu gösterir. Bir çizgi ile gösterilir. İstenirse çizginin altına açıklama yazılabilir. Multiplicity istenirse yazılabilir.

Bağlantı noktası (port): Birden fazla arayüzü, bir bileşene bağlarken gruplamakta kullanılır. Bileşenlere yapışık bir biçimde resmedilir. Aşağıdaki gibi bir dikdörtgen ile gösterilir. İstenirse içine isim yazılabilir.

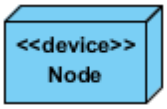
Akış (Deployment) Diyagramları

Akış diyagramları, sistemin yazılım parçalarını, onları çalıştıran donanımla eşleştirir; yazılım bileşenlerinin ve donanımın çalışma anı yapılandırmasında statik görünümünü gösterir; ve sistemin mantıksal elementlerini, onların fiziksel yerini ve birbirleriyle nasıl iletişim kurduğunu modeller.

Düğüm (node): Yapıları (artifact) çalıştıran, yazılımı içinde bulunduran fiziksel varlıklara bu isim verilir. Aşağıdaki gibi gösterilir. Kutunun içine düğümün ismi yazılır.

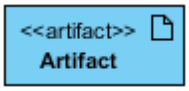


Aygıt düğümü (device node): Düğümün donanım olduğunu belirtir. Aşağıdaki gibi gösterilir. "<<device>>" yazısının altına düğümün ismi yazılır.

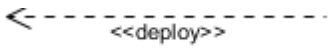


Yürütme ortamı düğümü (execution environment node): Yazılımı ifade eder. Aşağıdaki gibi gösterilir. "<<executionEnvironment>>" yazısının altına düğümün ismi yazılır.

Yapı (artifact): Sistemin bileşenlerinin çalıştırdığı fiziksel dosyalardır. Sistemin yazılımıyla ilgili bilgi parçalarıdır. Aşağıdaki gibi gösterilir. "<<artifact>>" yazısının altına yapının ismi yazılır.

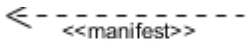


Yerleştirme (deploy): Bir bağımlılık stereotipidir. Bir yapının bir düğümde yer aldığını belirtir. Aşağıdaki gibi gösterilir. “<<deploy>>” yazısının altına istenirse bağımlılığın ismi yazılır.



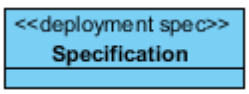
Bileşen (component): Sistemin yapıtaşlarıdır. Yazılımın yeniden kullanılabilen ve değiştirilebilen parçalarıdır. Aşağıdaki gibi gösterilir. “<<component>>” yazısının altına bileşenin ismi yazılır.

Görünme (manifestation): Bir yapının bir UML elementini uygulamasına denir. Bir bağımlılık stereotipidir. Aşağıdaki gibi gösterilir. “<<manifest>>” yazısının altına istenirse bağımlılığın ismi yazılır.



İlişki (association): Düğümler arasındaki ilişkidir. Bir çizgi ile gösterilir. İlişki istenirse okun üzerinde isimlendirilebilir.

Dağıtım Tanımlama (deployment specification): Yazılımın çalışması için gerekli olan konfigürasyon parametrelerini tanımlayan özel bir tür yapıdır. Aşağıdaki gibi gösterilir. “<<deployment spec>>” yazısının altına yapının ismi yazılır. Alttaki dörtgene istenirse dağıtım gereksinimleri yazılabilir.



Bağımlılık (dependency): Bir dağıtım tanımlamanın (deployment specification) bir yapıyı belirttiğini ifade eder. Kesikli bir ok ile gösterilir. Okun gösterdiği tarafta yapı, diğer tarafta dağıtım tanımlama yer alır. Bu gösterimde dilenirse, bağımlılık durumu okun alt tarafına yazılabilir. Bağımlılığın bir diğer anlamı da bir dağıtım tanımlamanın bir düğüme ait olduğunu göstermesidir. Bu gösterimde düğüm, okun gösterdiği taraftadır.