12. Hafta

YMÜ225-YMÜ321
Yazılım Gereksinim Analizi

Dr. Öğr. Üyesi Feyza Altunbey Özbay

İçerik

- Sınıf (Class) Diyagramları
- Sınıflar Arasındaki İlişkiler

Sınıf (Class) Diyagramları

Sınıf (Class); aynı metotları (işlev), aynı ilişkileri ve aynı anlamı paylaşan nesneler topluluğunun ortak tanımıdır. Sınıflar yazılımın durağan (statik) yapısının tanımlanmasında kullanılırlar.

Sınıf Diyagramları UML'in en sık kullanılan diyagram türüdür.

Sınıf diyagramı, sınıfların, arayüzlerin, ortaklıkların, işbirliklerinin ve kısıtlamaların bir koleksiyonunu gösterir. Yapısal diyagram olarak da bilinir.

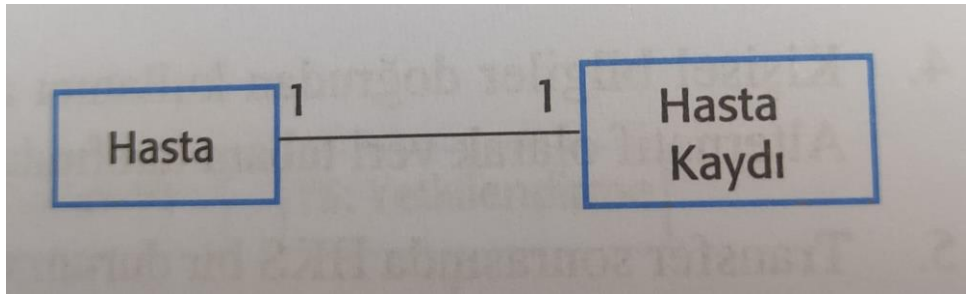
Sınıf (Class) Diyagramları

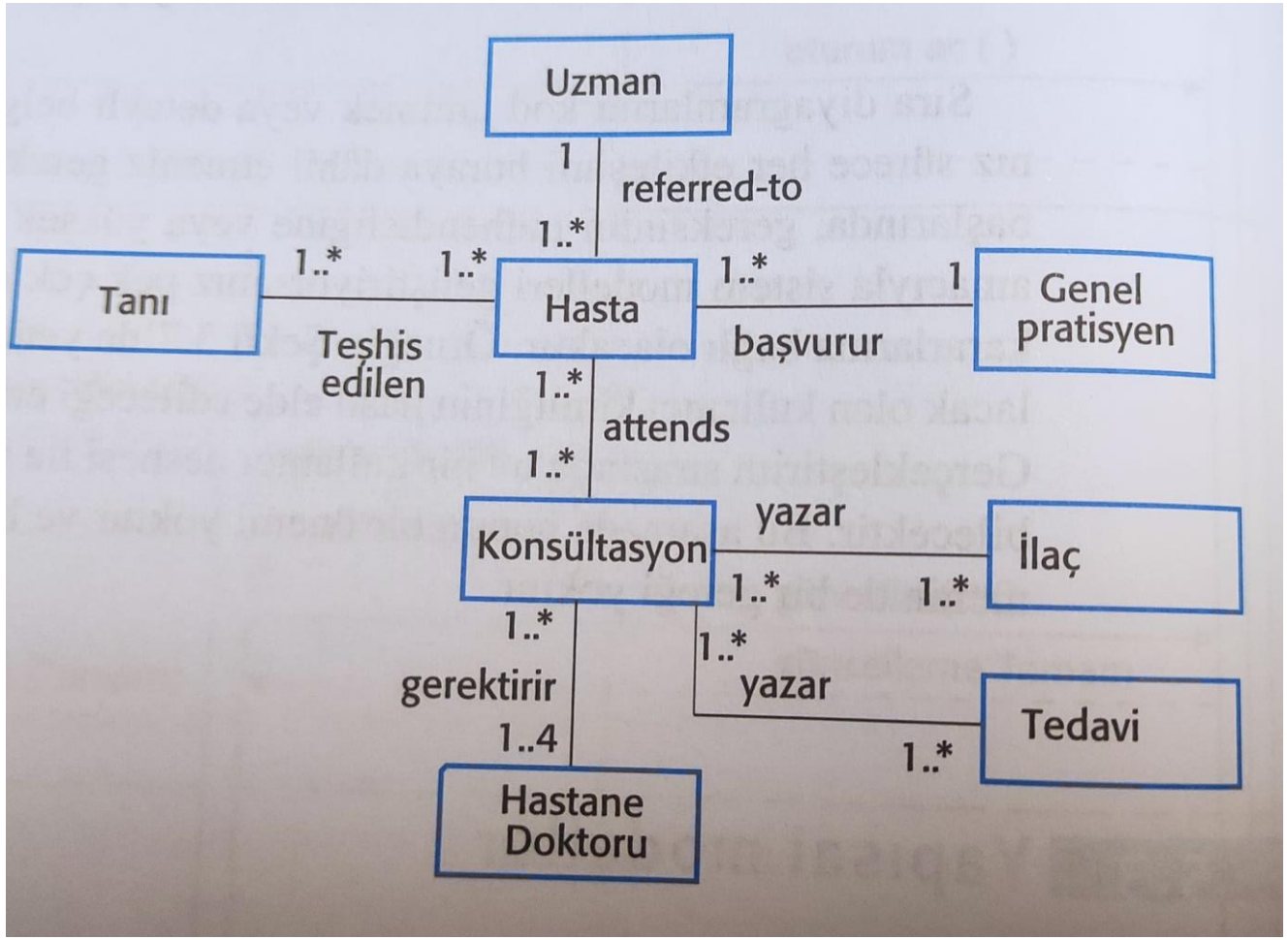
- Sınıf diyagramı statik bir diyagramdır.
- Bir uygulamanın statik görünümünü temsil eder.
- Sınıf diyagramı sadece bir sistemin farklı yönlerini görselleştirmek, açıklamak ve belgelemek için değil aynı zamanda yazılım uygulamasının yürütülebilir kodunu oluşturmak için kullanılmaktadır.
- Sınıf diyagramı bir sınıfın niteliklerini ve işlemlerini ve ayrıca sistem üzerinde uygulanan kısıtlamaları açıklar.
- Sınıf diyagramları **nesne odaklı dillerin** modellemesinde yaygın olarak kullanılmaktadır, çünkü bunlar doğrudan nesne yönelimli dillerle eşleştirilebilen tek UML diyagramlarıdır.

Sınıf (Class) Diyagramları

Sınıf (Class) Diyagramının amaçları:

- Bir uygulamanın statik görünümünün analizi ve tasarımı.
- Bir sistemin sorumluluklarını açıklayın.
- Bileşen ve dağıtım diyagramları için temel.
- İleri ve geri mühendislik.


Hasta
1
1
Hasta Kaydı

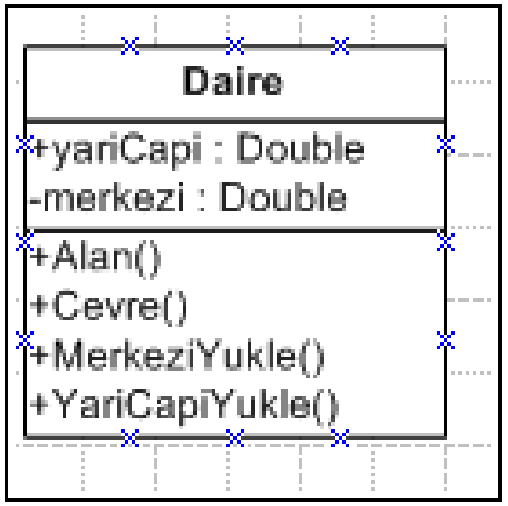

Uzman
1
referred-to
1..*
Tanı
1..*
1..*
Hasta
1..*
1
Genel pratisyen
Teşhis edilen
başvurur
1..*
attends
1..*
Konsültasyon
yazar
İlaç
1..*
1..*
1..*
1..*
gerektirir
yazar
1..4
Tedavi
1..*
Hastane Doktoru

Sınıf (Class) Diyagramları

Sınıfların;

- Bir adı
- Niteliklerin (Attributes) adı
- Metot / İşlevleri
- Bunlara ek olarak
  - Constraints (koşullar)
  - Notes (Notlar)

vardır.



Constraints (koşullar) sınıfa ilişkin bir takım koşulların belirtildiği ve parantez içinde yazılan bilgilerdir.

Notes (Notlar) genellikle işlevlerin ve özelliklerin hakkında bilgi veren opsiyonel kutucuklardır.

Sınıf (Class) Diyagramları

Sınıf diyagramlarında yer alan nitelik ve metot isimlerinin önünde bazı bezemeler (adornments) bulunmaktadır.

- **Public (+):** Nitelik ya da metot genel kullanıma açıktır, diğer sınıflardan da erişilebilir.
- **Protected (#):** Nitelik ya da metota sınıf içerisinden ve kendisinden türeyen alt sınıflar (subclasses) tarafından erişilebilir.
- **Private (-):** Nitelik ya da metota yalnızca içinde bulunduğu sınıf tarafından erişilebilir.

Sınıflar Arasındaki İlişkiler

Sınıf diyagramları kendi başlarına son derece yalın yapılardır ve görsel olarak çizilmeleri, ancak aralarındaki ilişkilerle birlikte incelediklerinde anlamlıdır. UML içerisinde sınıflar arasında 4 değişik tür ilişki tanımlanabilir:

- Bağıntı ilişkisi (Association)
- Genelleme ilişkisi (Generalization)
- Bağımlılık ilişkisi (Dependency)
- Gerçekleştirim ilişkisi (Realization)

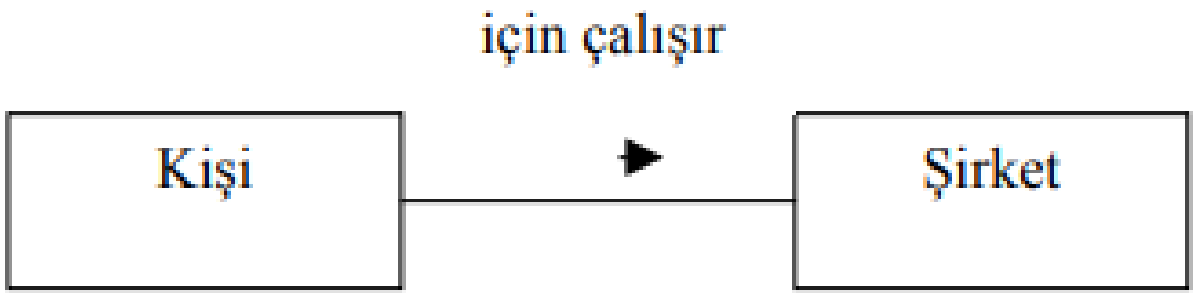
Bağıntı İlişkisi (Association)

- Bağıntı ilişkisi, bir sınıfa ait nesnelerin diğer sınıfa ait nesnelere nasıl bağlandığını tanımlar ve sınıflar arasında düz bir çizgi ile gösterilir.

- Bağıntı türü ilişkiler için tanımlanmış bilgiler;
  - Bağıntının adı
  - Sınıfın bağıntıdaki rolü
  - Bağıntının çokluğu

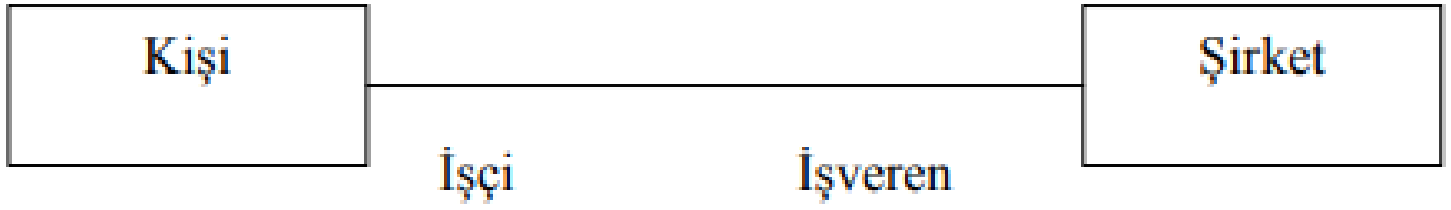
Bağıntı İlişkisi (Association)

Bağıntı adı, iki sınıf arasındaki ilişkinin küçük bir açıklamasıdır. Bu açıklama yazılırken gerekirse yön bilgisi de içi dolu küçük bir üçgen ile gösterilebilir.

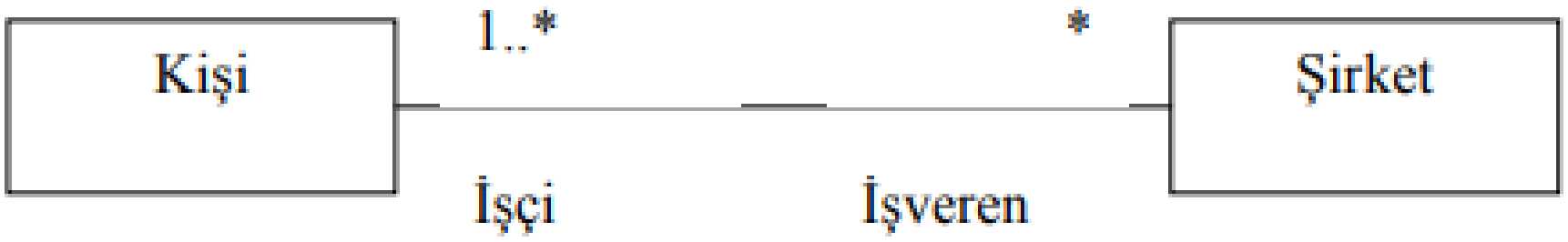


Bağıntı İlişkisi (Association)

İki sınıfın nesneleri arasındaki ilişkide her sınıfın nesnesinin üstlendiği **rol**ün de belirtilmesi, sınıf diyagramlarının daha kolay anlaşılmasını sağlar.



Bağıntı İlişkisi (Association)



- Yukarıdaki gösterimde işçi-veren ilişkisinde bir şirketin en az bir işçisi olduğu (1..*), bir işçinin is o ya da herhangi bir sayıda (*) şirkette işçi olarak çalışmış olabileceği ifade edilmektedir.

Bağıntı İlişkisi (Association)

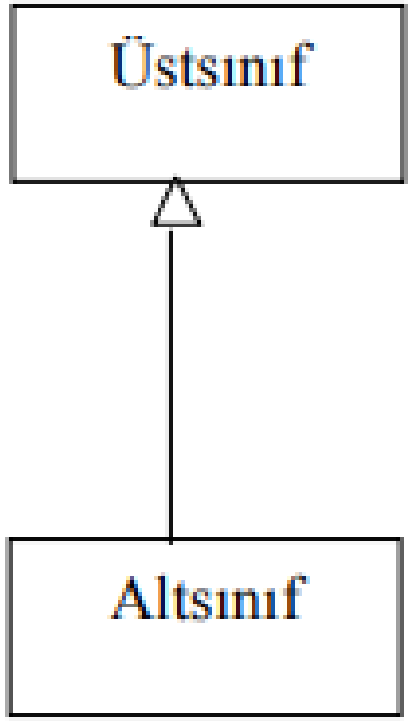
En temel bağıntı ilişki tipleri aşağıda listelenmiştir:

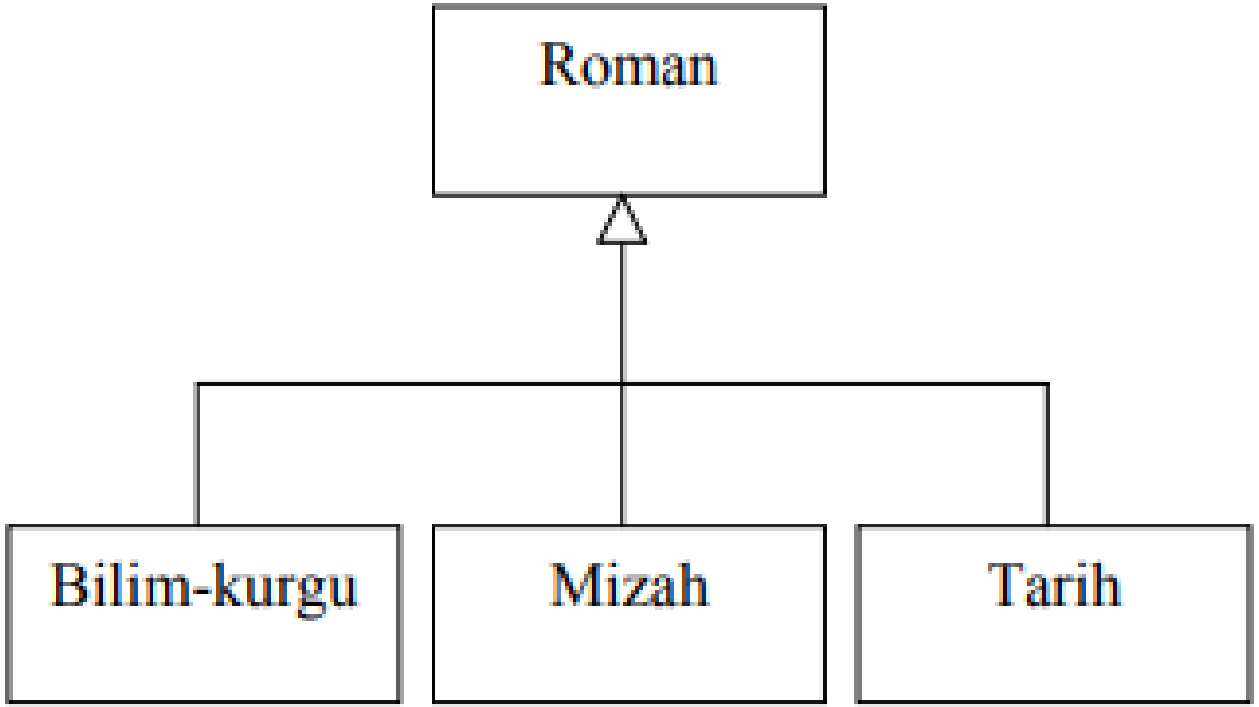
Çokluk Bezemesi	Açıklama
1	Yalnızca 1 (herhangi bir sayma sayısı da kullanılabilir)
0..1	Yalnızca 0 ya da 1
M..N	En az M, en çok N
*	0 ya da daha çok
0..*	0 ya da daha çok
1..*	1 ya da daha çok

Genelleme İlişkisi (Generalization)

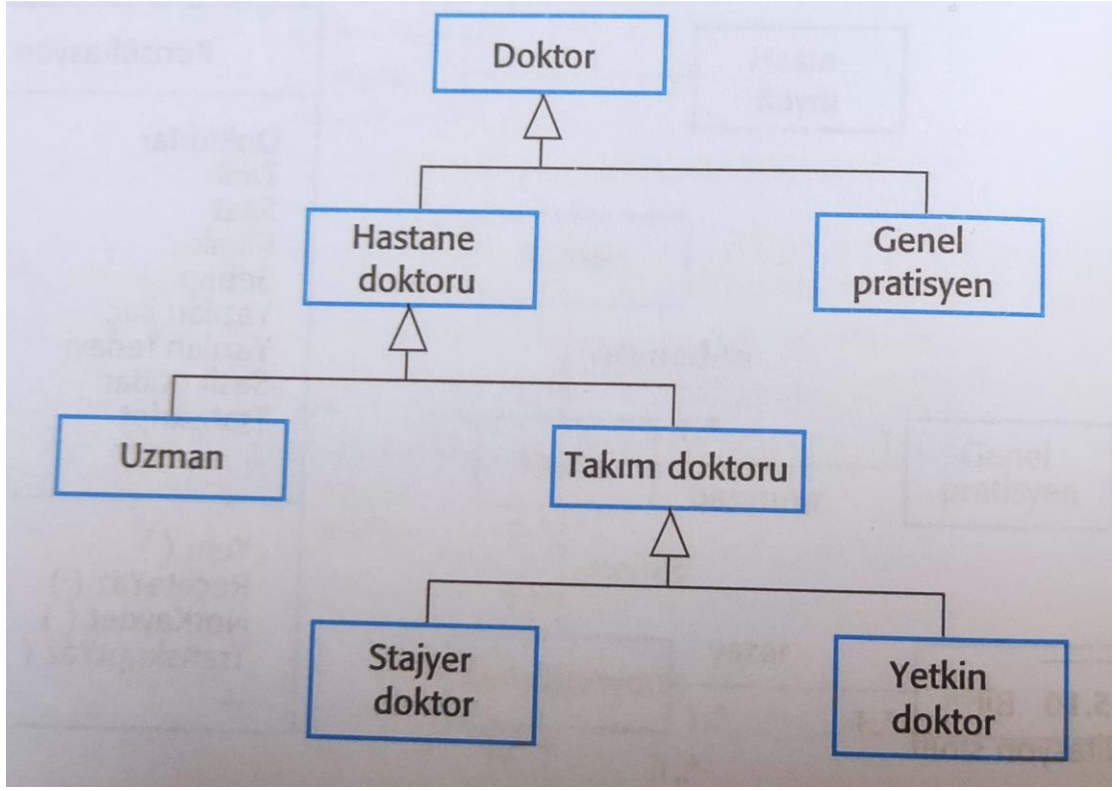
- Bu ilişki, bir nesnenin bir diğerinin özel bir türü olduğunu modellemek için kullanılır.
- Bu ilişkide nesneler genelden özele doğru bir dizilim ile yerleştirilir.
- Genelleme ilişkisi sıradüzensel bir yapı oluşturduğundan sınıflar arasında üst-sınıf (super-class) ve alt-sınıf (sub-class) ayrımı yapılır.
- Alt sınıflar içi boş bir ok ile üst sınıfa bağlanır.

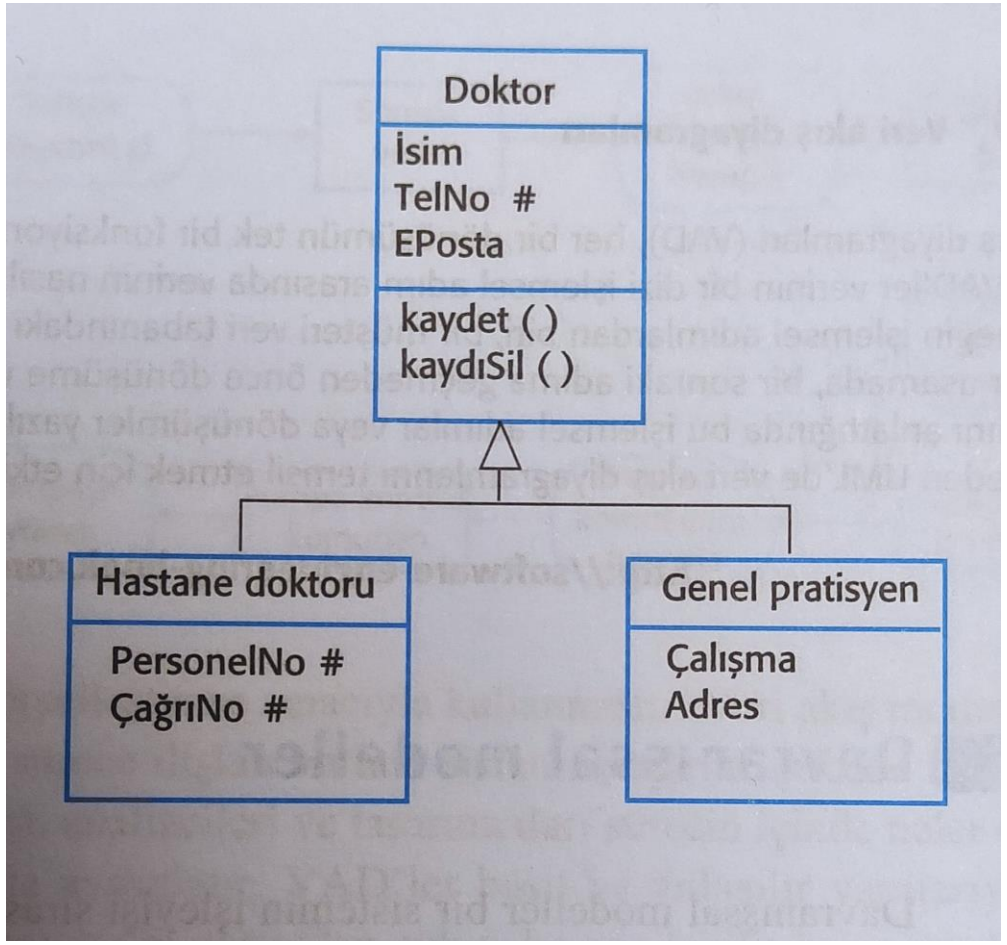
Genelleme İlişkisi (Generalization)





Genelleme İlişkisi (Generalization)





Kalıtım (Inheritance)

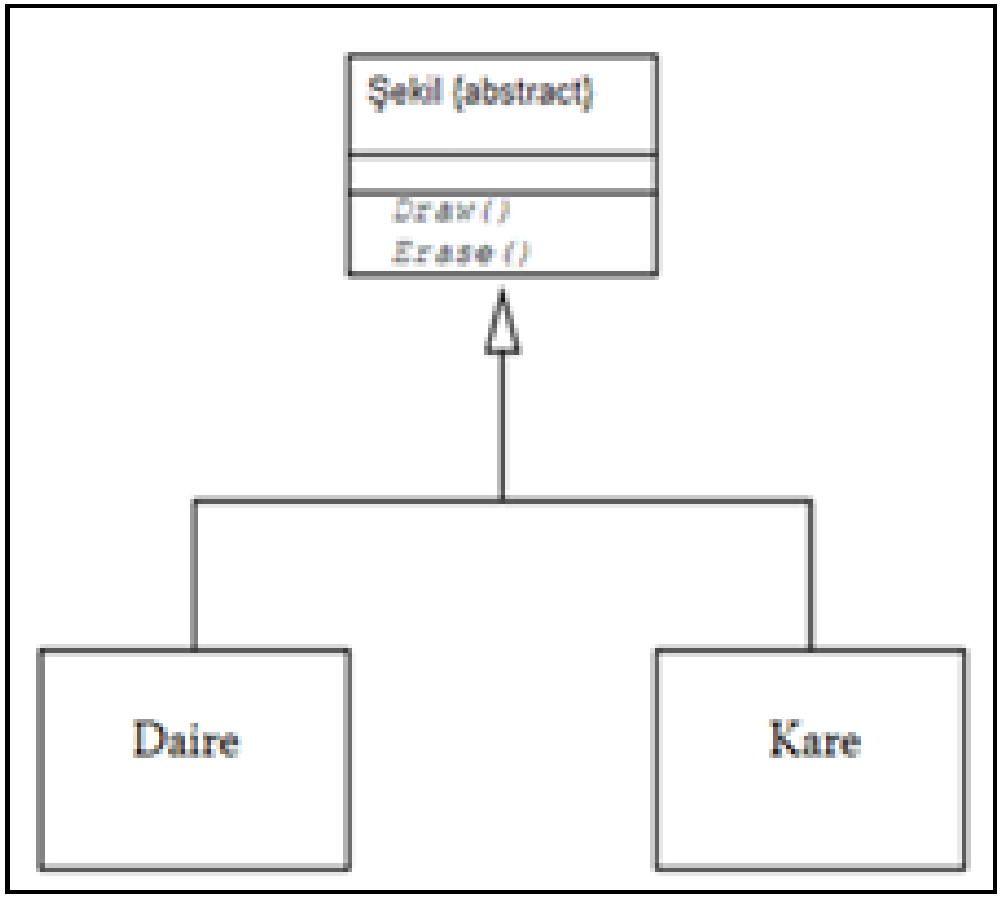
Genelleme ilişkisi nesneye yönelik programlama dillerinde kalıtım (inheritance) mekanizması ile gerçekleşir.

Türetme yoluyla bir sınıf başka bir sınıfın var olan özelliklerini alarak, o sınıf türünden başka bir nesneymiş gibi kullanılabilir.

Bir sınıfın işlevleri türetme yoluyla genişletilecekse, türetmenin yapılacağı sınıfa taban sınıf (base class), türetilmiş olan sınıfa da türemiş sınıf (derived class) denir.

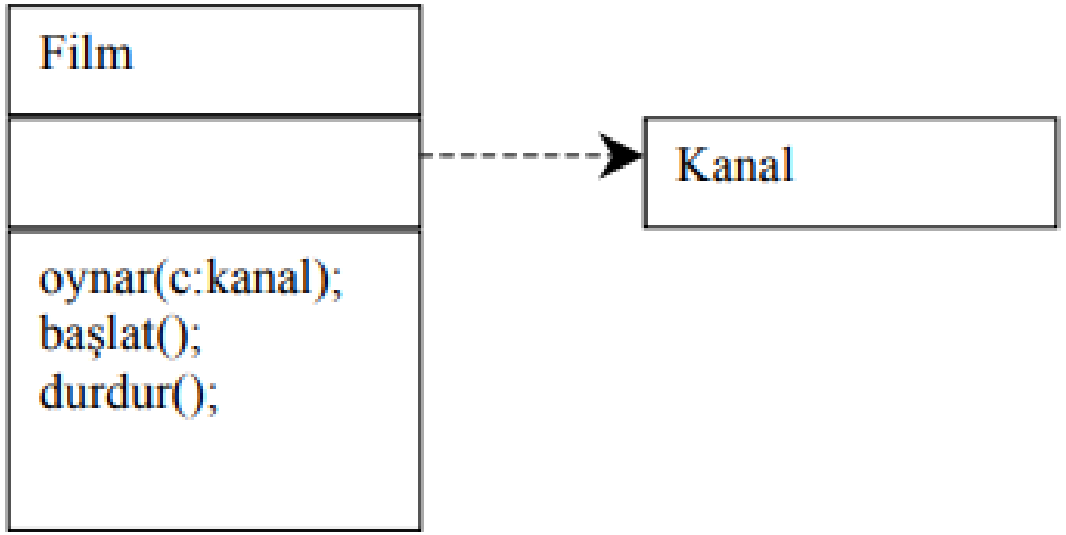
Kalıtım (Inheritance)

- Türetme ağacında daire ve kare birer şekildir.
- Bu durumda "Şekil" sınıfı taban sınıf (base class), daire ve kare ise türetilmiş (derived class) sınıflardır.
- Türetmeye sınıflar arası ilişki açısından baktığımızda türetmenin "is kind of" (bir çeşit) ilişkisinin olduğu görülür.



Bağımlılık İlişkisi (Dependency)

Bağımlılık ilişkisi, «kullanır» ilişkisinin modellenmesinde kullanılır. Kullanır ilişkisinde bağımlı (dependent) ve bağımsız (indepent) ayrımı bulunur. Eğer bir sınıf, diğer sınıf metotlarından en az birisinde parametre olarak kullanılıyorsa iki sınıf arasında bağımlılık ilişkisi vardır. Bağımlılık ilişkisi, bağımlı sınıftan bağımsız sınıfa doğru kesikli çizgi ile çizilen bir ok ile gösterilir.



Film sınıfı hangi kanalda gösterildiğine dair bir metoda sahiptir. Bu nedenle, *Film* sınıfı, *Kanal* sınıfına bağımlıdır. *Kanal* sınıfında yapılacak bir değişim *Film* sınıfında da değişiklik yapılmasını gerektirirken, tersi söz konusu değildir.

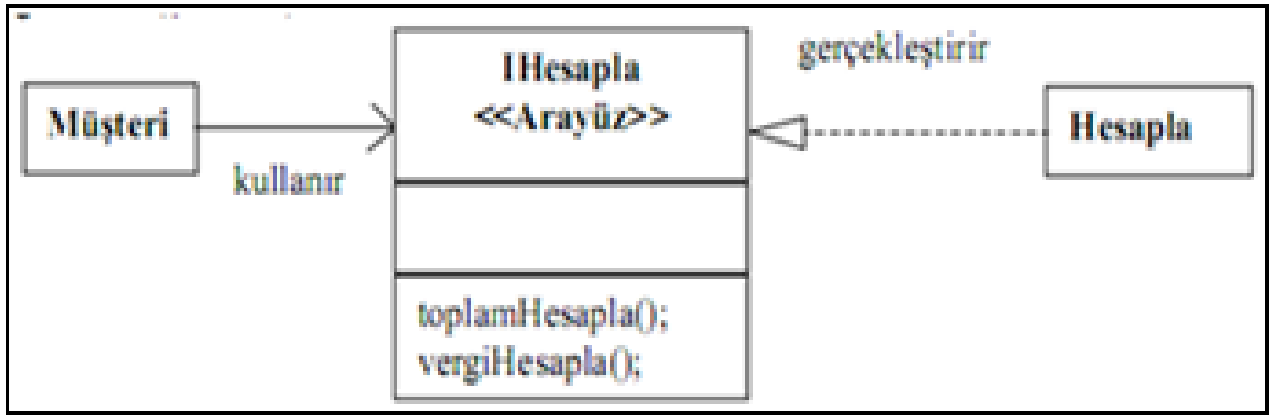
Gerçekleştirim İlişkisi (Realization)

Gerçekleştirim ilişkisi en çok kullanıcı arayüzlerinin (user interface) modellenmesinde kullanılır.

Arayüz yalnızca method adlarını ve bunların parametrelerini içermektedir.

Gerçekleştirim ilişkisi kesikli bir çizginin ucuna yerleştirilen içi boş bir üçgen ile gösterilir.

Gerçekleştirim İlişkisi (Realization)



- IHesapla adı verilen ve diğer sınıfların kullanımına açılmış arayüz sınıfı tanımlanmıştır.
- Bu sınıfta yer alan metotların gerçekleştirimi ise Hesapla sınıfı tarafından üstlenilmiş durumdadır.
- İyi yapılan bir tasarımda tüm sınıflar yalnızca arayüzde yer alan metotları kullanmalı, doğrudan Hesapla sınıfının metotları çağırılmamalıdır.
- Böylece Hesapla sınıfının gerçekleştirimi, uygulamanın kalanını etkilemeksizin istenildiği gibi değişebilir.