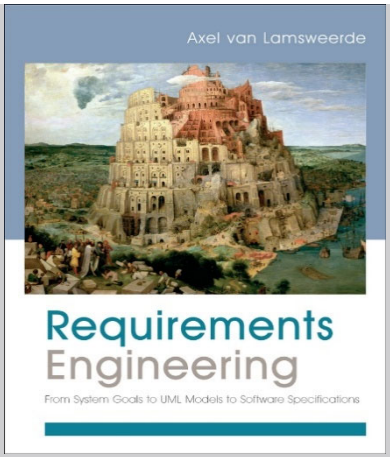


Sistem Modellerinin Oluşturulması

Building System Models for RE

Bölüm 10

Kavramsal Nesnelerin Sınıf Diyagramları ile Modellenmesi

Modeling Conceptual Objects with Class Diagrams

Kavramsal Nesnelerin Sınıf Diyagramları ile Modellenmesi

- Kavramsal modelleme ile uygulama alanına özel bildirimler yapılır.
- Bu bildirimler gerçekleştirilen çözüme göre sistem ile ilgili «prescriptive» ya da «descriptive» ifadeler olabilir.
- Böylece gereksinimler mühendisliğinin ilk çalışmaları olan yapısal (structural) modelin bu aşamadaki uygulamaları gerçekleştirilir.
- Nesne modelinin kullanımı birtakım «entities, associations, agents, events» şeklinde tanımlanacaktır.
- Bunlar «varlıklar, birliktelikler, agents, olaylar» şeklinde de ifade edilebilir.
- Nesne modelleme UML sınıf diyagramları notasyonlarını kullanan «entity-relatioship» diyagramlarla görselleştirilmiştir

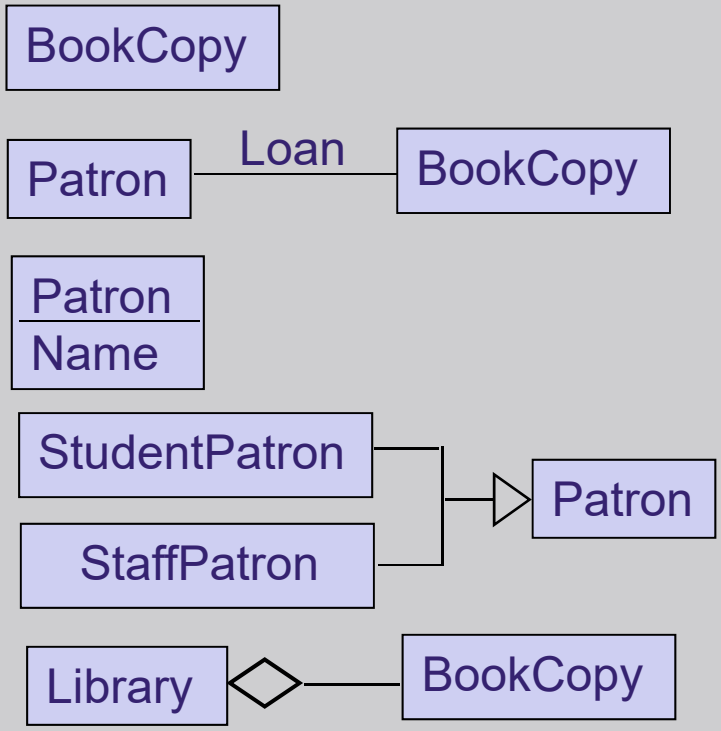
Kavramsal Nesnelerin Alan özellikleri ile Modellenmesi

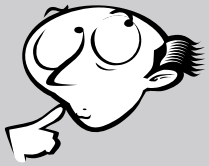
Kavramsal nesne nelerden oluşur?

- Varlıklar (Entities)
- Birliktelikler ve Çokluklar

(Associations & multiplicities)

- Öznitelikler (Attributes)
- Özelleştirme(Specialization)
- Birleştirme (Aggregation)
- Sınıflar üzerindeki diğer bağıntılar
  - AND-associations , OR-associations





Kavramsal Nesne Nedir?

- Kavramsal Nesne sistemin özel bir kavramına ait örnekler kümesidir
  - Kavramsal Nesnelerin her biri bağımsız olarak tanımlanır
    - string veri tipinin genel tanımlamasında iki farklı string için "Justine Henin" verildiğinde; bu iki örnek veri tipi anlamında aynıdır. OYSA
    - Bir Patron nesnesine ait iki farklı örnek (instance) Justine Henin olarak verilebilir.
    - Bu iki örnek birbirinden ayırt edilebilir. Çünkü bunlar farklı nesnelerdir. İçeriklerindeki değerler farklıdır.
  - Sistemin herhangi bir durumunun listelenmesi mümkündür
    - Herhangi bir durumda iken Patron nesnesinin tüm örnekleri mevcut durumda sistemde bulunan kavramların tümünü vermek üzere listelenebilir.

Kavramsal Nesne Nedir?

- Kavramsal Nesne ile benzer özellikler paylaşılır (share similar features)
  - Ortak olan **isim, tanımlama, tip, alan özellikleridir (**name, definition, type, domain properties)
  - Ortak olan **öznitelikler(attributes) , birlikteliklerdir. (associations)**

Örneğin: Patron nesnesinin email isimli bir özniteliği (attribute) vardır. Patron nesnesi ile BookCopy nesneleri arasında birliktelik (association) bağlantısı olduğu için bu öznitelik paylaşılır.

- Kavramsal bir nesne örnekleri ile (instances) birlikte sisteme ait bir olguyu (phenomena) oluşturur.

-

Kavramsal Nesne Örneği

- Bir öznitelik herhangi bir nesnenin farklı durumlarına özel olabilir.
  - Buradan da bir öznitelik (attribute) aynı nesnenin diğer örneklerinde (instances) farklılıklar gösterebilirler.

Örneğin: Sistemin herhangi durumunda aynı nesnenin farklı örneklerinin ortak öznitelikler ve birliktelikler (associations) için farklı değerleri vardır.

Farklı trenlerin aynı sistem durumunda *Speed* özniteliği *ON* birlikteliği için farklı değerler içerebilir.

Kavramsal Nesnelerin Oluşturulması -I
Davranışsal Amaçların betimlenmesi

- Sistemin gerçekleştireceği davranışların tümü «davranışsal amaçlar» olarak tanımlanır. Bunlar aracıların-temsilcilerin (agents) paralel olarak gerçekleştirecekleri davranışlardır.
- Durum (state), davranışları simgeler. Durum bir x değişkeninin v değeri şeklinde (x,v) fonksiyon çifti olarak ifade edilir.
- Davranışsal Amaç: «Tren hareket halinde iken tüm kapılar her zaman kapalı kalacaktır» örneği 2 durum değişkeni içerir:

  «trainMovement» ve «trainDoors».

Olası durumlardan biri (trainDoors, 'closed') fonksiyon çifti ile betimlenir. Bu fonksiyon çifti iki alt durumu birleştirir:

trainMovement ve trainDoors

Kavramsal Nesnelerin Oluşturulması Davranışsal Amaçların Betimlenmesi

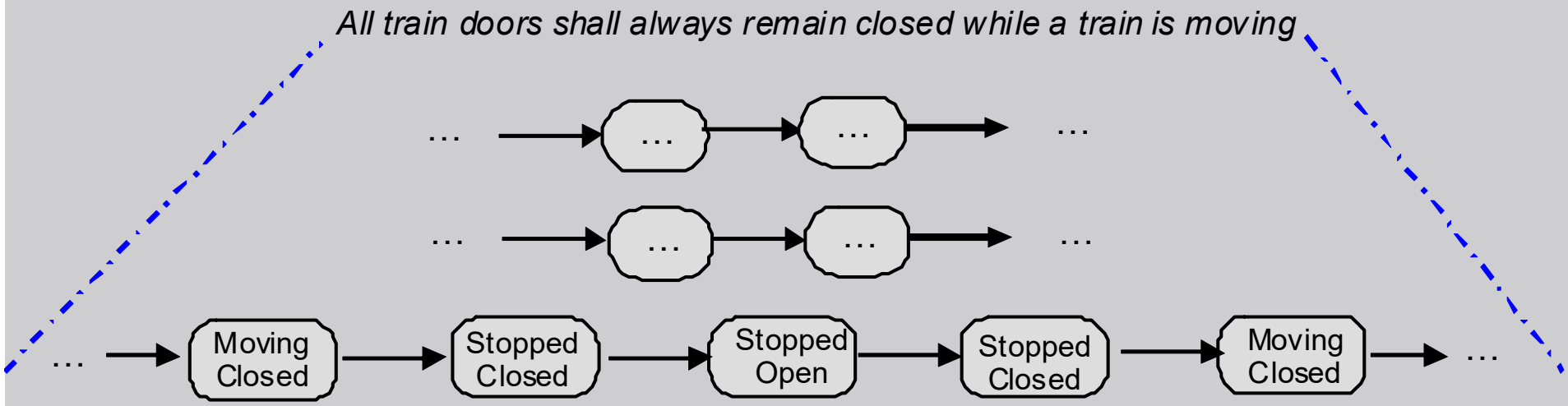


Figure 7.3 – Behavioral goals prescribe intended system behaviors

Kavramsal Nesnelerin Oluşturulması
Davranışsal Amaçlar: Gerçekleştirme Amacı

- Gerçekleştirme amaçlı bir davranışsal amaç (achieve goals) bir amacın eninde sonunda (sooner or later) gerçekleştirilmesini hedefler. Diğer koşullar mevcut sistem davranışını sürdürmeye devam ederler.

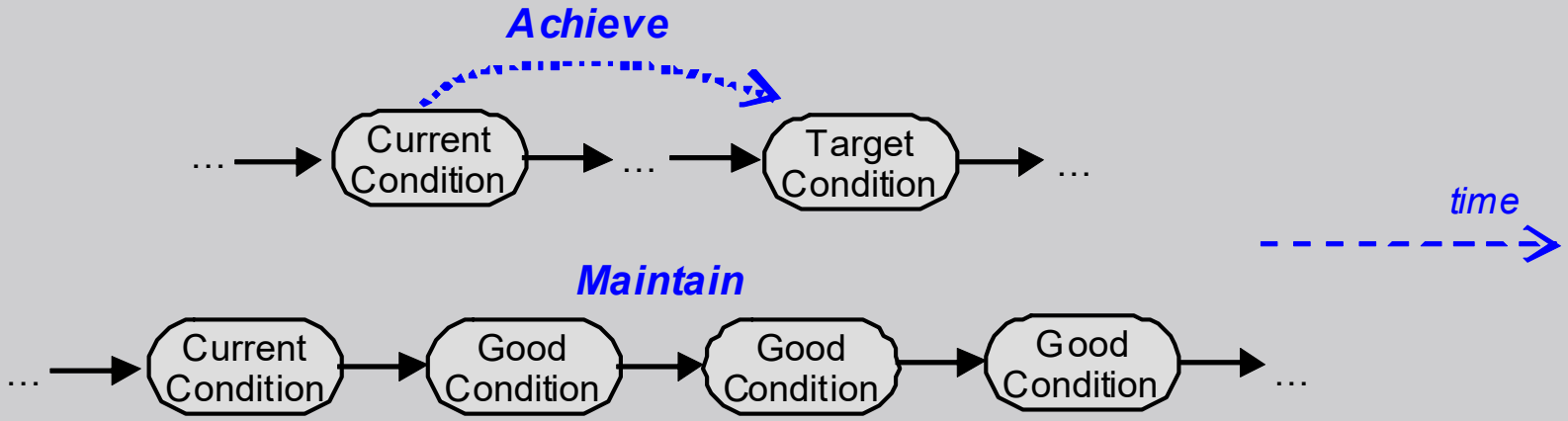
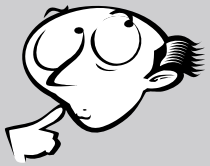


Figure 7.4 – Behavioral goals: *Achieve* and *Maintain* goals

Tren bir platformdayken, eninde sonunda (sooner or later) sonraki platforma geçecektir

Kitap ödünç almak üzere rezerv edildiğinde eninde sonunda alıcıya ödün verilecektir.

Bir toplantı yapılması planlandığında , eninde sonunda herkesin katılımı ile gerçekleşecektir.



Kavramsal bir nesnenin örneğinin bir durumu nedir?

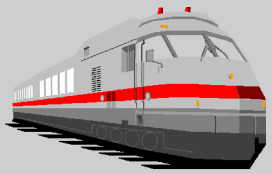
What is a state of an instance of conceptual object ?

- Fonksiyonel çiftlerin kaydı $x_i \mid\rightarrow v_i$

  x_i : nesne özniteliği (attribute) ya da birlikteliği (association)

  v_i : Bu örneğe karşılık gelen değer şeklinde ifade edilir.

Train nesnesinin *tr* örneği aşağıda ifade edilen durumda tanımlanmaktadır.

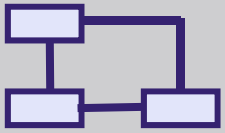


(tr.Speed $\mid\rightarrow$ 0, tr.Location $\mid\rightarrow$ 9.25, tr.DoorsState $\mid\rightarrow$ Open,

On $\mid\rightarrow$ (tr, block13), At $\mid\rightarrow$ (tr, platform1))

Train objesinin farklı bir durumunun *tr' örneği olacaktır.*

Block:tren otomasyon sisteminde her tren hattı (trace –iz) eşit uzunlukta parçalara ayrılır. Her bir parçaya «block» adı verilir ve her bir «block» içerisinde sadece tek bir tren gidebilir. Bu çarpışmayı önlemek amaçlı bir bölünmedir. İki tren arasındaki en kısa mesafe koşulu güvenli seyehat için yeterli değildir.



Nesnenin Örneklenmesi -I (Object instantiation)

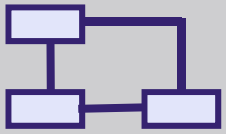
Bir kavramsal nesnenin, hangi örneklerinin mevcut nesnenin elemanı olduğunu anlatan yerleşik (built-in) bir semantik ilişki tanımlaması gerekir:

InstanceOf (*o*, Ob) **sadece ve sadece** *o , Ob nesnesinin örneği olduğunda geçerlidir.*

- Mevcut durum = sistemin seçilmiş olan herhangi bir durumu
- **InstanceOf (*bc*, BookCopy)** bildiriminde *bc, kütüphane sistemindeki kitap kopyalarını veren* BookCopy nesnesinin bir elemanıdır (örneğidir-instance).

- Nesnenin örnekleri belli bir süre için, yani süreç süresince (phenomena olarak da belirtilebilir) geçerlidir.
- Bir nesne örneği bir nesneden diğer nesneye geçiş yapabilir.

StudentPatron örneği (**instance**) → StaffPatron örneği (**instance**)



Nesnenin Örneklenmesi –II (Object instantiation)

- Nesne modelde her kavramın anlamsal bir ilişkisi vardır. Her örneğin sağlaması gereken gerek ve yeter şartları temsil eder.

  InstanceOf (*o*, Ob) **şeklinde ifade edilir.**

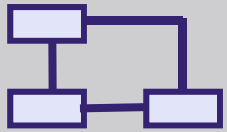
  - Bu nesnenin örneği olarak olması gereken tüm koşullar (nesne tarafından görülebilecek ve görülemeyecek) verilmelidir.
  - Örneğin: " *patron* bir kişidir. Kütüphaneye belli bir süre için kayıtlıdır. Bu tarihten sonra kütüphaneden yararlanamaz.

- Bir nesnenin örneği verildiğinde, nesnenin öznitelikleri ve birliktelikleri (associations) **durum değişkenleri** (state variables) olarak başlatılır. Böylece nesnenin örneği karakterize edilmiş olur.

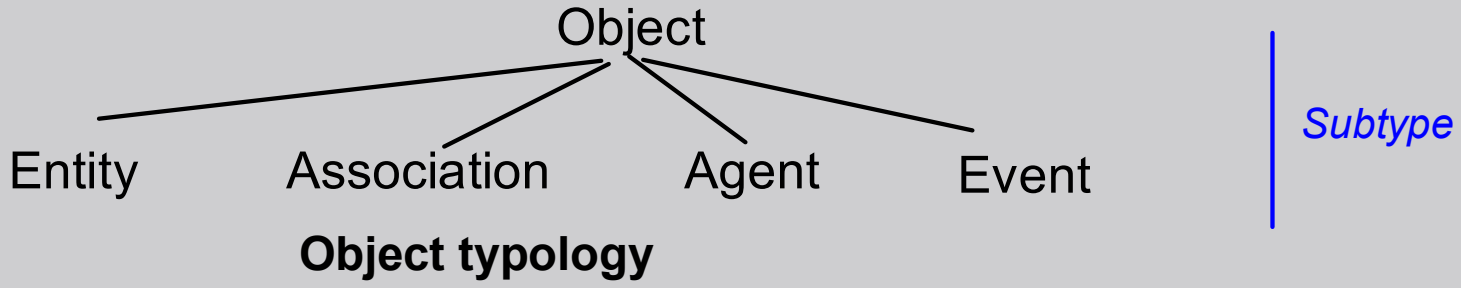
  InstanceOf (*tr*, Train) → tr.*Speed*, tr.*DoorsState*, *On* (tr, ...)

- Sistemin durum değişkenleri = nesne modelinde bildirilmiş olan tüm kavramsal nesnelerin durum değişkenlerinin kümesidir.

Kavramsal Nesne Tipleri -I



- Varlık (Entity) Nesnesi: bağımsız (autonomous) ve pasif (passive) nesnedir.
  - Bir sistemin örnekleri (instances) diğer nesnelerin örneklerinden bağımsız olabilirler.
  - Varlık nesnelerinin örnekleri **diğer nesnelerin davranışını kontrol etmek zorunda değildir**.
  - Örneğin: Book, BookCopy ve Train, Platform, Block nesneleri varlık (entity) nesneleridir. Bu nedenle diğer nesnelerin davranışlarını kontrol etmezler.
  - Varlık (entity) nesneleri , UML sınıf diyagramı ile gösterilir.



Object typology

Kavramsal Nesne Tipleri - II

- Birliktelik (Association): Başka nesneye bağlantısı olan kavramsal bir nesne belirtilir.
- Örnekler (instances), birbirleri ile ilişkili nesne örnekleri arasındaki kavramsal bağlantılardır.
  - **Loan** bir birliktelik (association) linkidir ; Patron olarak tanımlanan agent ile BookCopy varlığını (entity) birbirine bağlar.
  - **Copy** bir birliktelik (association) linkidir; BookCopy ve Book varlıklarını (entities) birbirine bağlar.
  - **At** bir birliktelik (association) linkidir; Train & Platform varlıklarını (entities) birbirine bağlar.
  - **On** bir birliktelik (association) linkidir; Train & Block nesnelerinin bağlantısıdır..

Bunlar, UML gösteriminde association olarak betimlenir.

Kavramsal Nesne Tipleri - III

- **Agent** bağımsız (autonomous) ve aktif bir nesnedir.
- Örnekleri nesnenin tek tek davranışlarıdır. Bu davranışlar, nesnenin her bir örneğinin (instance) kontrol ettiği bir dizi durum değişkenleri ile gerçekleşir. Örneğin :
- TrainController isimli *agent* örneği Train örneklerinin (instances) hız özniteliğini (attribute) kontrol edebilir. Bu, bir davranışı etkilemek demektir.
- Staff *agent* kavramsal nesnesi ile ilgili bir örnek Loan birlikteliğindeki diğer kavramsal nesnenin örneklerini (instances) kontrol edebilir; yani davranışları üzerinde değiştirici etkisi vardır.

Kavramsal Nesne Tipleri - IV

- Olay (Event) anlık bir nesnedir.
- Bir olay (event) örneği sistemin tek bir durumunda oluşur. Örneğin:
- Tren kontrol sisteminde StartTrain olayının (event) bir örneği örneklenen tren hareket ettiğinde oluşacaktır. Oysaki önceki durumda tren durmakta idi.

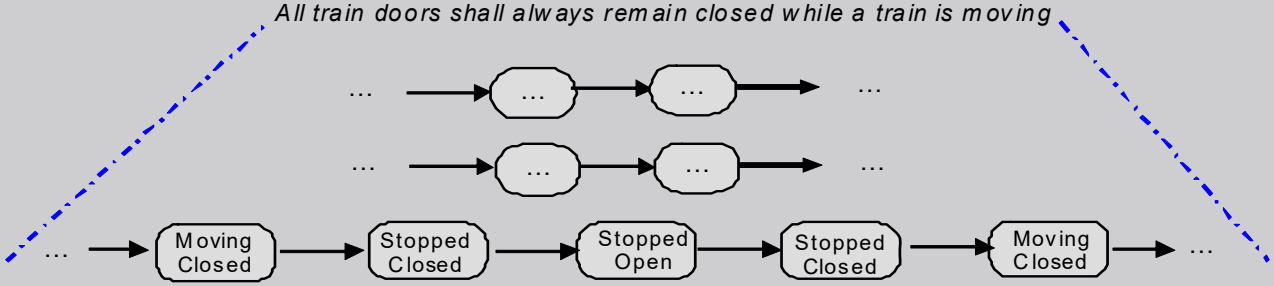
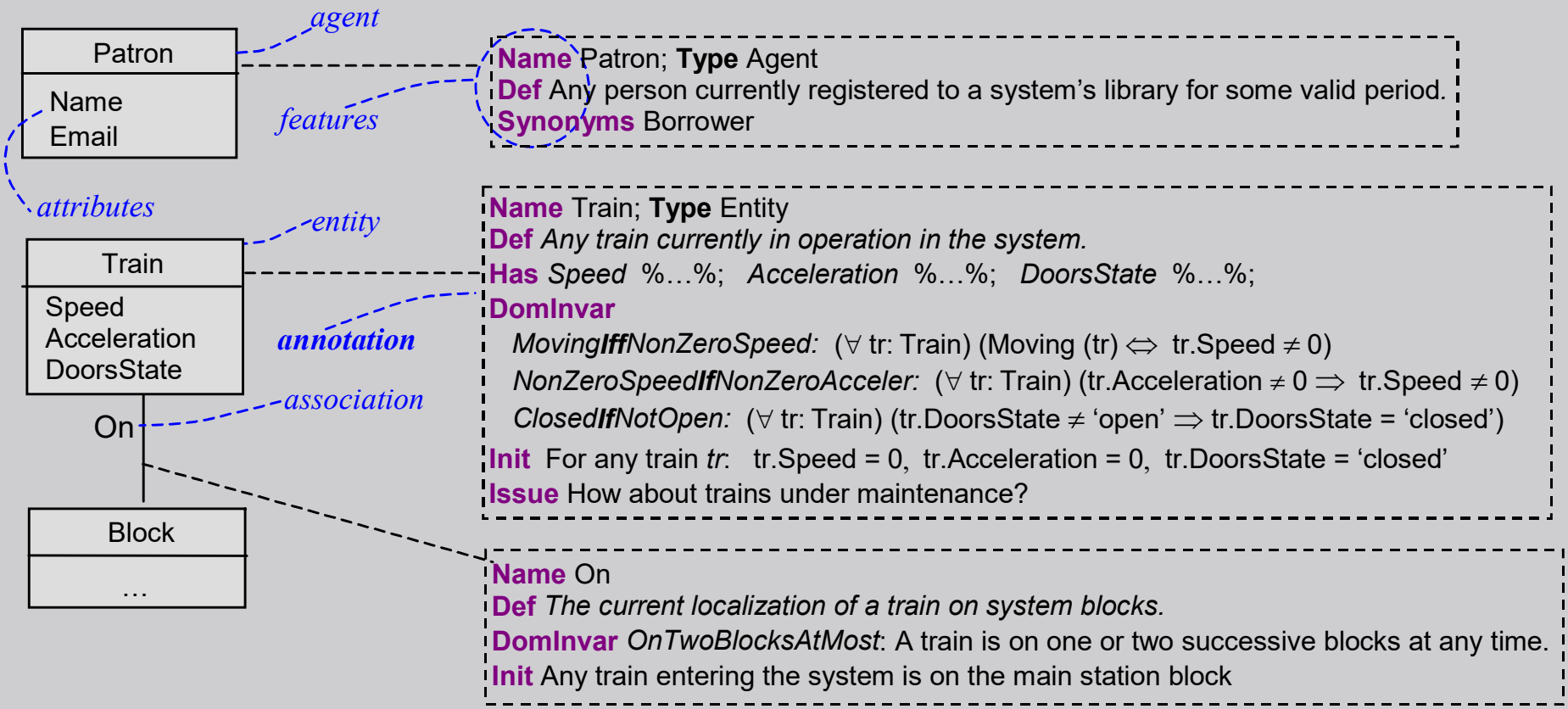


Figure 7.3 – Behavioral goals prescribe intended system behaviors

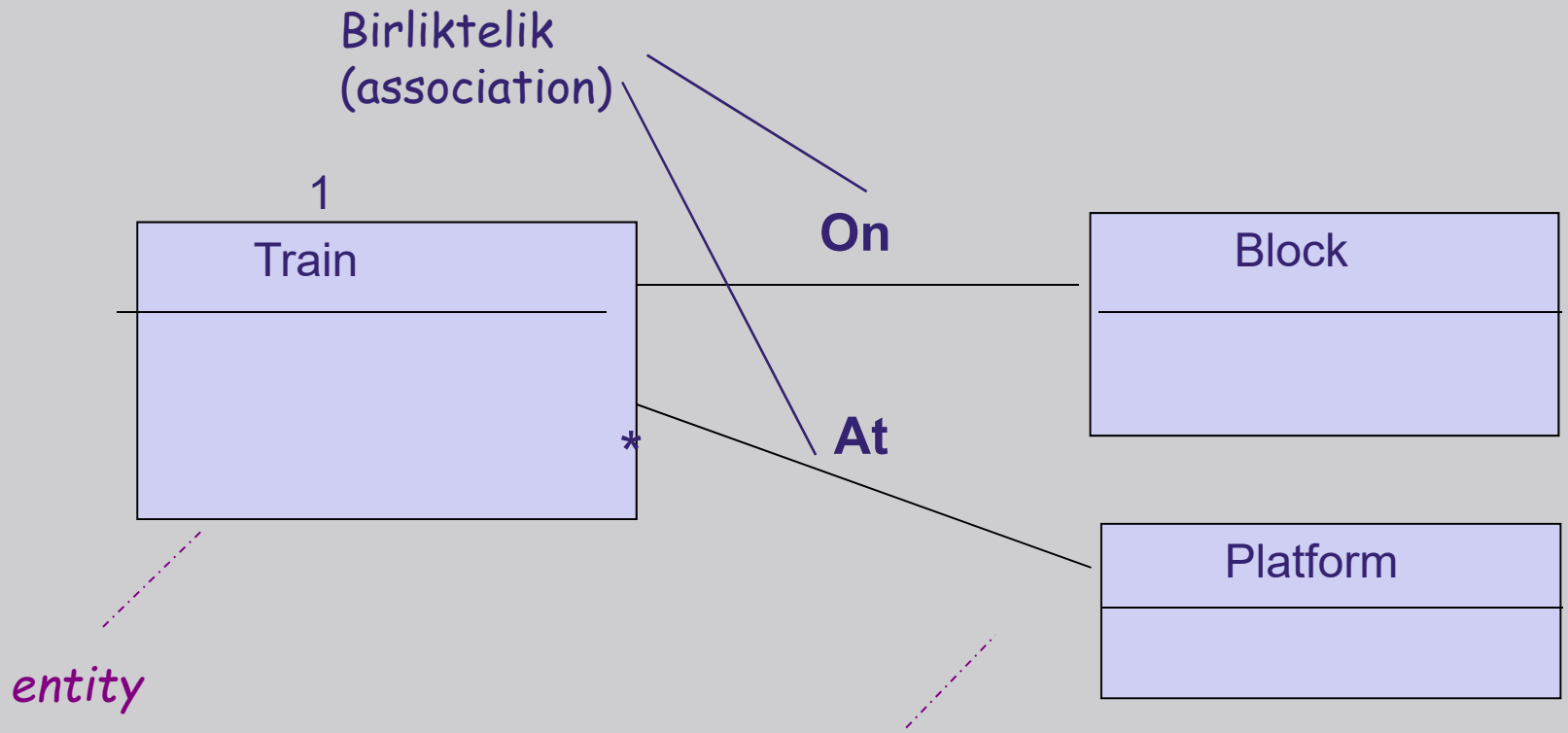
- BookRequest olayının (event) bir örneği, herhangi bir *agent* için bir kitap ödünç alma isteminde bulunulmasıdır.
  - Örnekler *tek bir sistem durumu* için mevcut olduğu için:

    InstanceOf (ev, Ev) , Occurs (Ev) şeklinde biçimsel olarak ifade edilir.

Modeli Açıklayan Nesne Özellikleri
Object features as Model Annotations

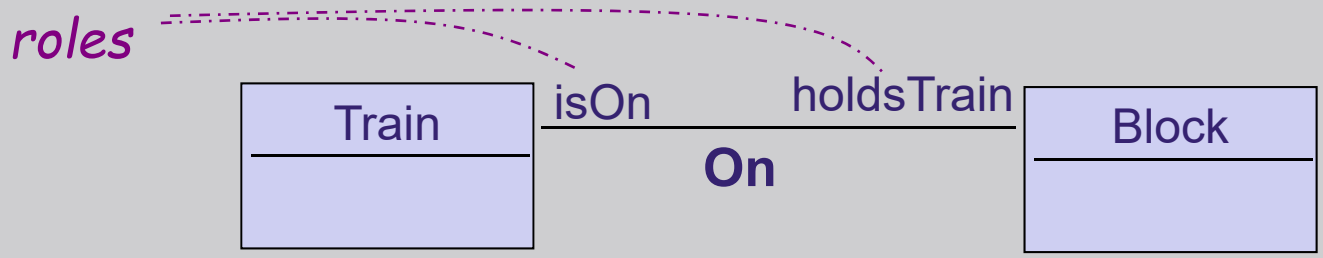


UML de Varlıklar ve Birliktelikler (Entities, Associations in UML)

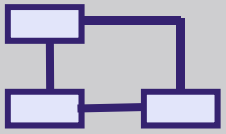


Kavramsal Nesne Association (Birliktelik)

- Association = Kavramsal bir nesnedir ve diğer nesnelerle bağlantısı vardır. Her bir nesnenin farklı bir işlevi vardır .
  - İlişki kurulan nesneye bağlıdır.
  - Bağlı nesneler varlıklar (entities) , birliktelikler(association) event(olay) , ya da agents olabilir.

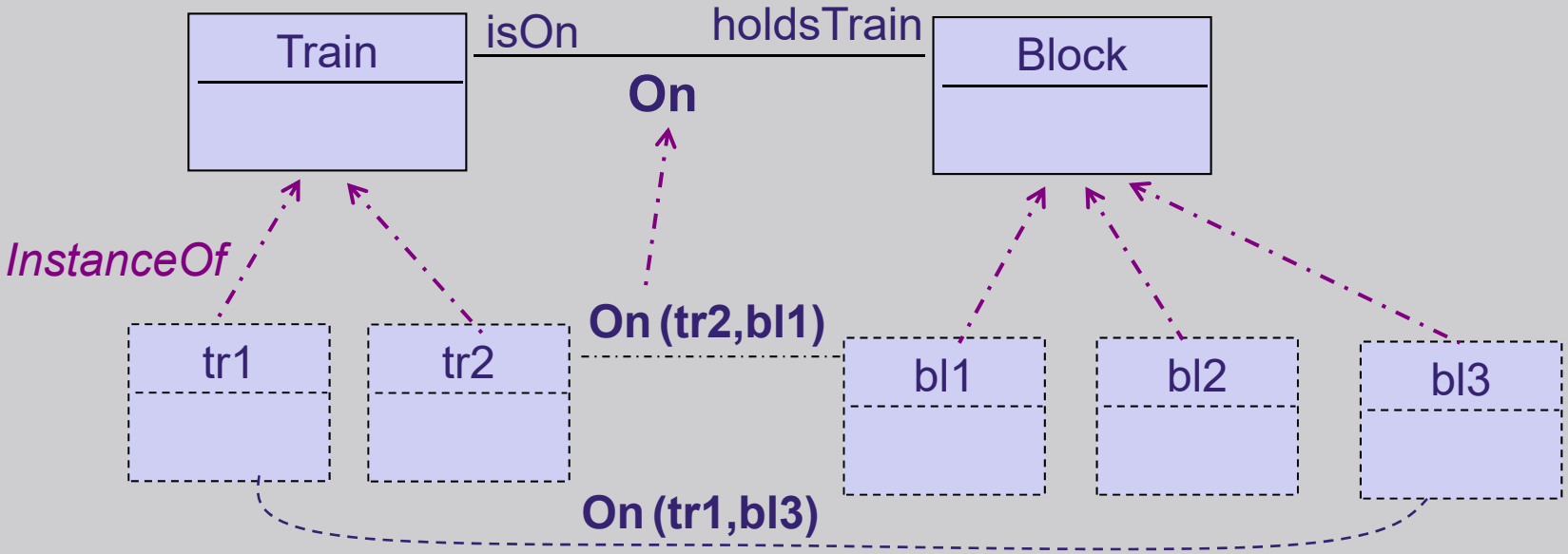


On birlikteliği (association) *Train* varlık (entity) nesnesi için isOn rolüne sahiptir.
Block varlık (entity) nesnesi ise On birlikteliğinde holdsTrain rolünü gerçekleştirir.

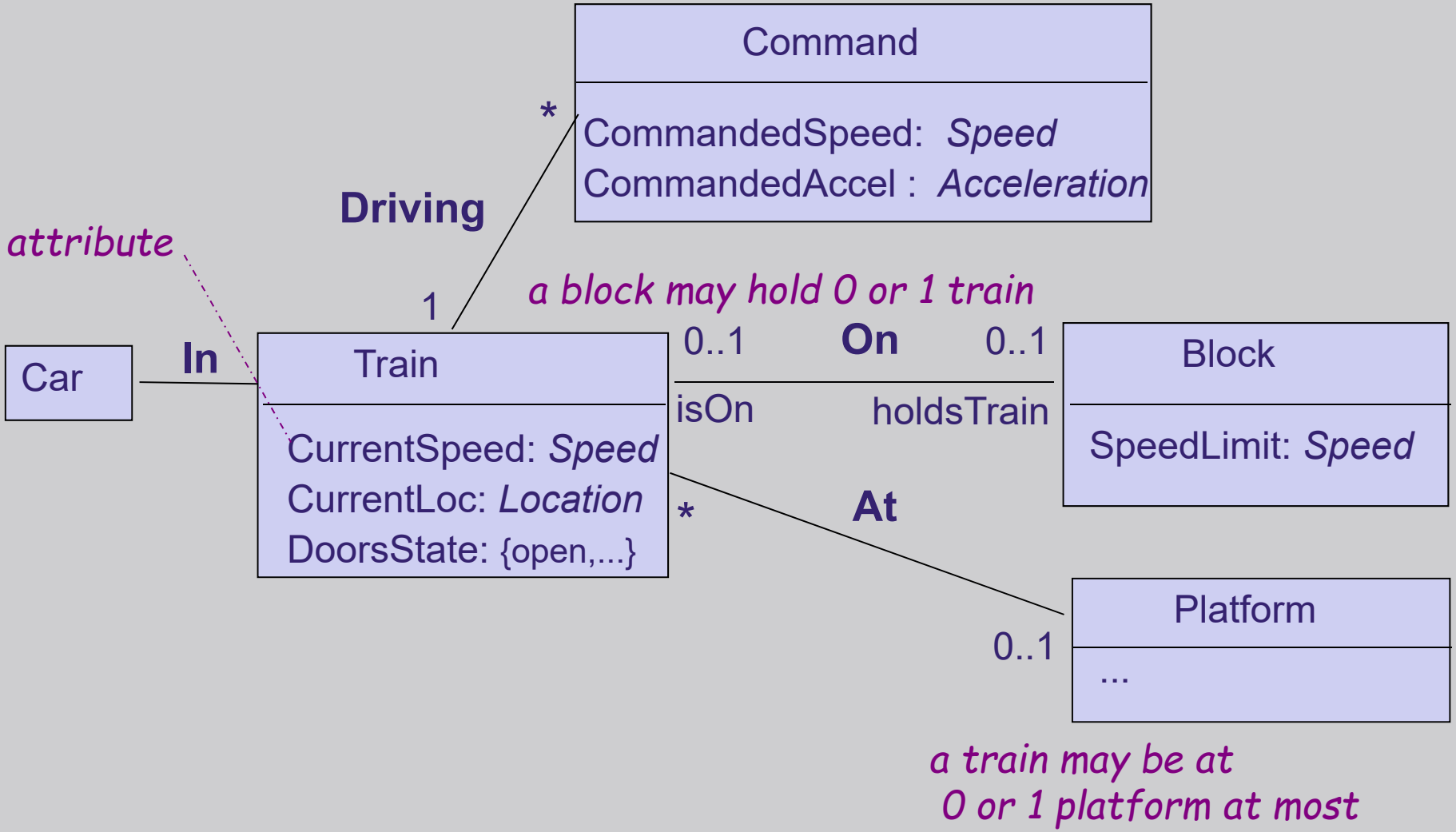


Birliktelik Örnekleri (Association Instances)

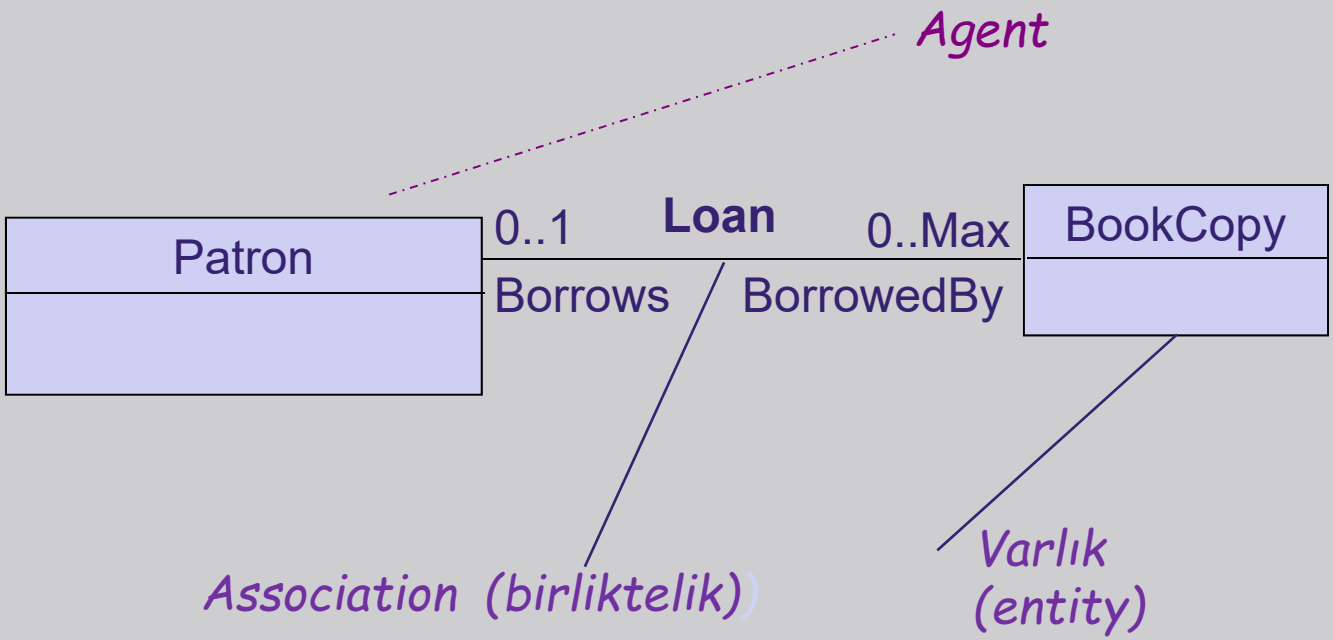
- Birliktelik örneği: = Bağlı nesnenin örnekleridir ve her birinin kendine ait bir rolü vardır

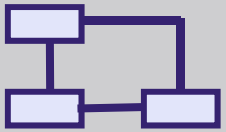


Varlıklar, Birliktelikler, Öznitelikler
Entities, Associations, Attributes in UML



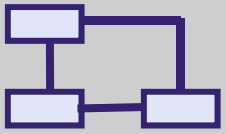
UML de Varlıklar, Agents ve Birliktelikler
Entities, agents, associations in UML





Birlikteliklerin büyüklükleri (Multiplicities of Association)

- İkili birliktelikler için kısıtlarla ilgili standartlar:
  - min = 0: seçmeli (optional)link (bazı durumlarda bağlantı yoktur)
  - min = 1: zorunlu (mandatory) link (herhangi bir durumda en az lir bağlantı olmalıdır)
  - max = 1: teklik (uniqueness) (herhangi bir durumda en fazla bir bağlantı hedeflenir)
  - max = *: hedef (target) örnekleri için keyfi (sıfır veya daha fazla) bir sayıdır.
  - "k" gösterimi "k..k içindir ",
  - "*" gösterimi "0..*" içindir



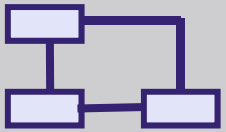
Büyüklükler, Alan özellikleri ve Amaçlar
Multiplicities, domain properties and goals

- Büyüklükler (multiplicities) aşağıdaki bildirimleri belirler.
  - domain properties (descriptive)

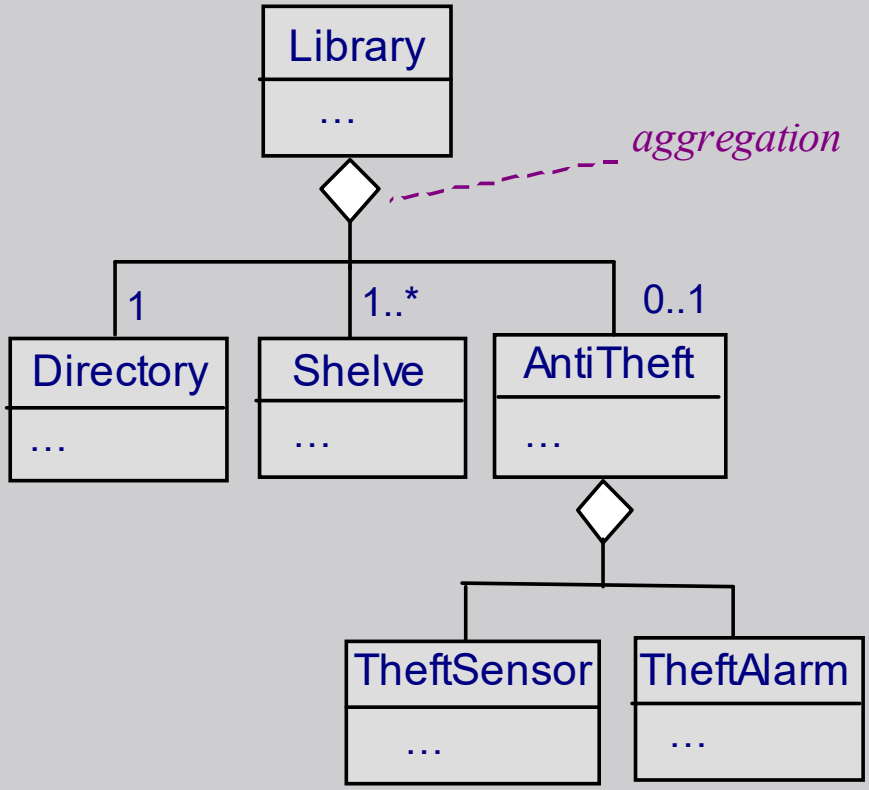
    «Tren belli bir zamanda en fazla tek bir platformda olabilir."
  - goals (prescriptive)

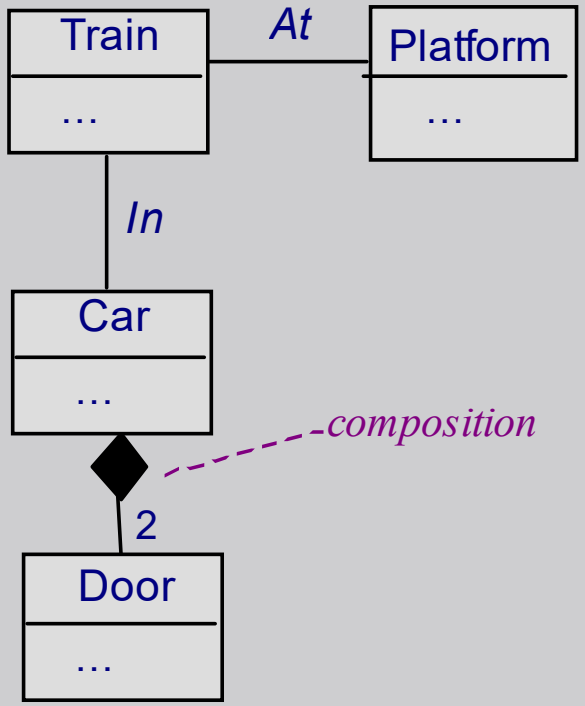
    «Bir blok herhangi bir zamanda birden fazla treni bulunduramaz."

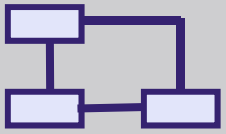
    «Bir kişi bir seferde maksimum kitap kopyalarından fazlasını alamaz."



Object Aggregation & Object Composition

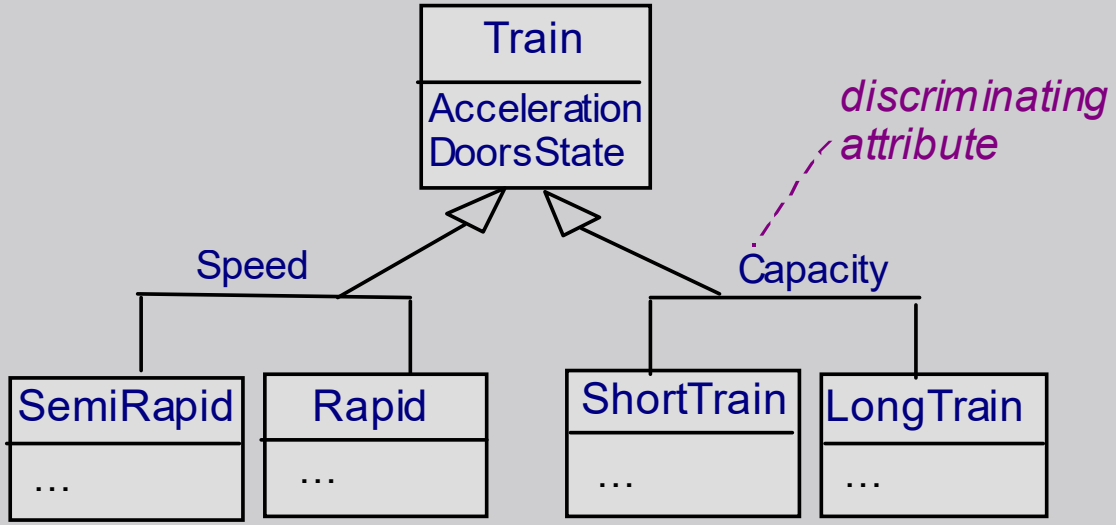


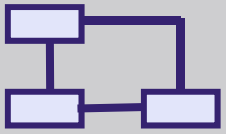




Çoklu Özelleştirme
Multiple specializations

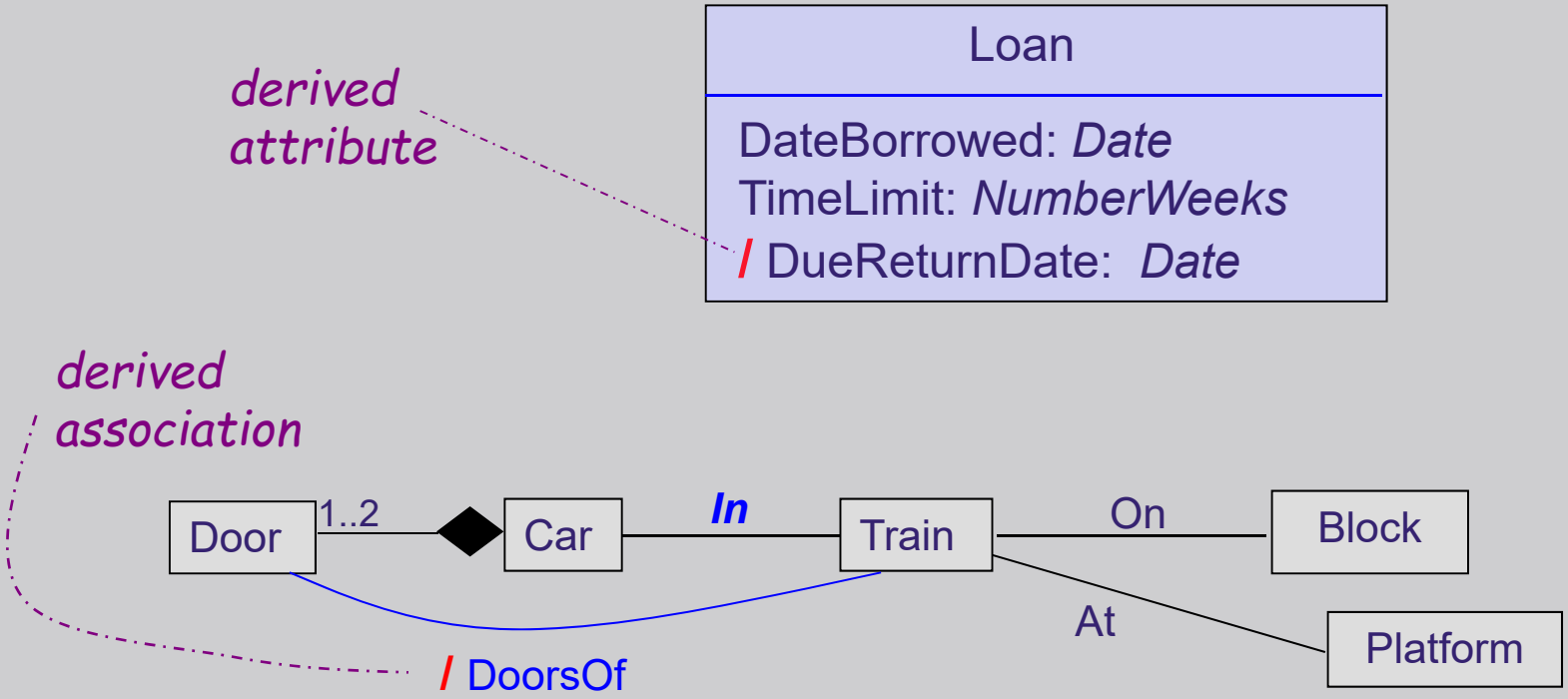
- Same object may have multiple specializations
  - Different subsets of object instances associated with different criteria
  - Same object instance may be member of different subsets (one per criterion)
- Discriminator = attribute of super-object whose values define different specializations (differentiation criterion)

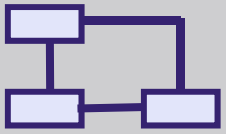




More on UML class diagrams

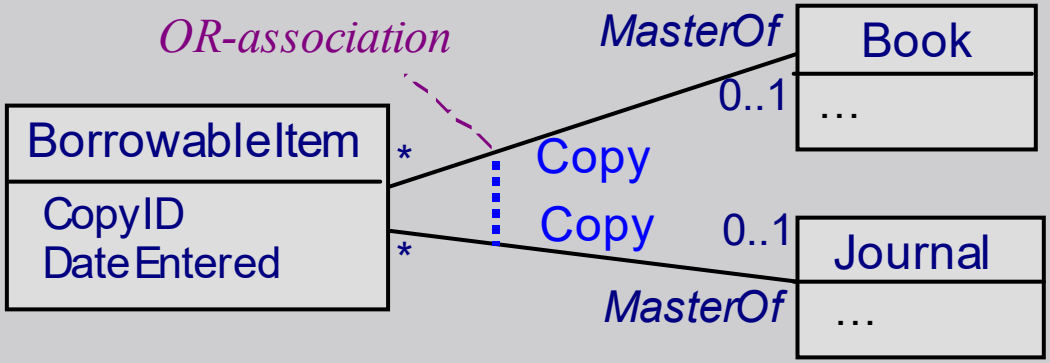
- Derived attribute, association = defined in terms of other attrib/assoc already in the model
  - controlled form of redundancy





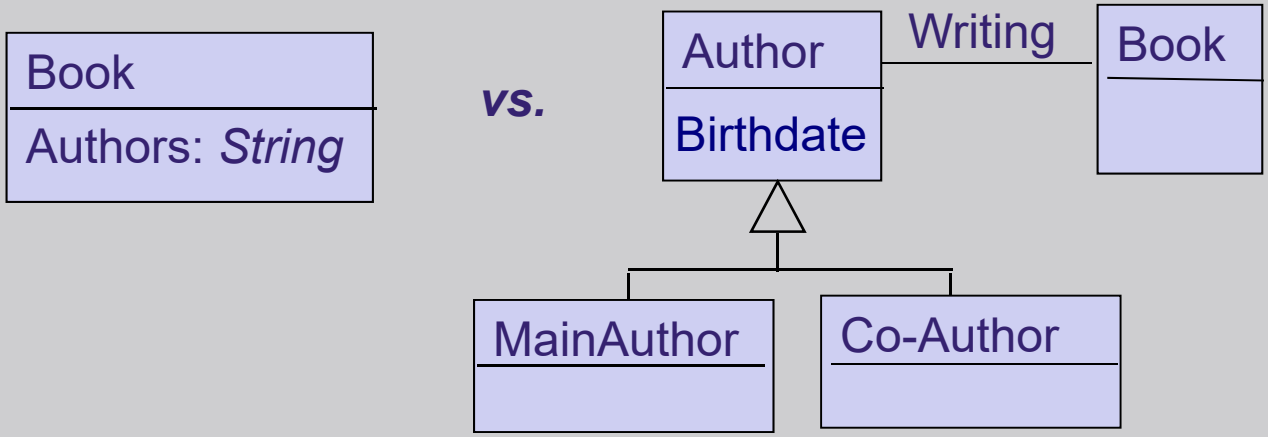
More on UML class diagrams

- OR association = same role played by alternative objects
  - set of object instances in this role =
    union of alternative sets of object instances



Object or Attribute ?

- For *X*: conceptual item in goal specs, make *X* an attribute if...
  - *X* is a function: yielding one single value (possibly structured) when applied to conceptual instance
  - instances of *X* need not be distinguished
  - you don't want to attach attributes/associations to *X*, specialize it, or aggregate/decompose it
  - its range is not a concept you want to specialize or attach attributes/associations



Entity, association, agent, or event ?

- For *X*: conceptual object in goal specs ...
  - instances of *X* are defined in one single state
    ⇒ event *e.g.* StartTrain
  - instances of *X* are active: control behaviors of other object instances
    ⇒ agent *e.g.* DoorsActuator
  - instances of *X* are passive, autonomous
    ⇒ entity *e.g.* Train
  - instances of *X* are passive, dependent on other, linked object instances
    ⇒ association *e.g.* Following (Train, Train)
      - *N*-ary if each of the N parties ...
        - need be considered as objects
        - yields tuples to be distinguished