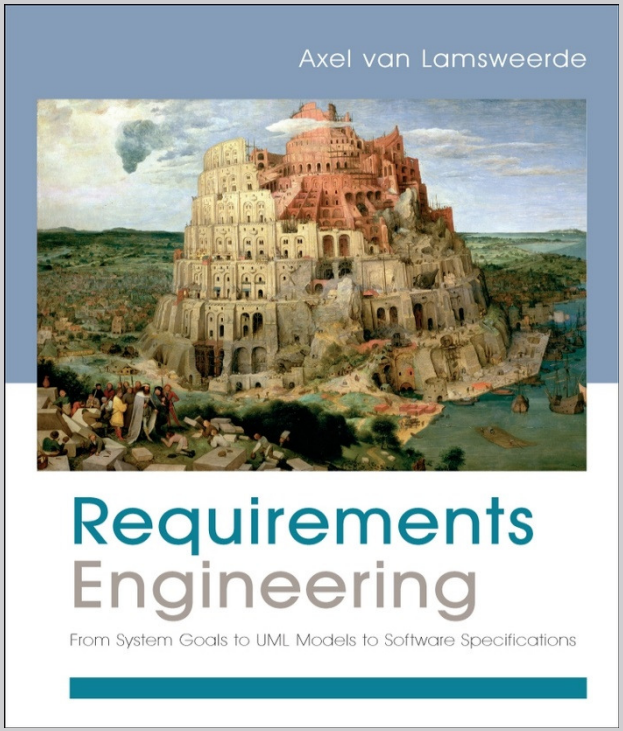
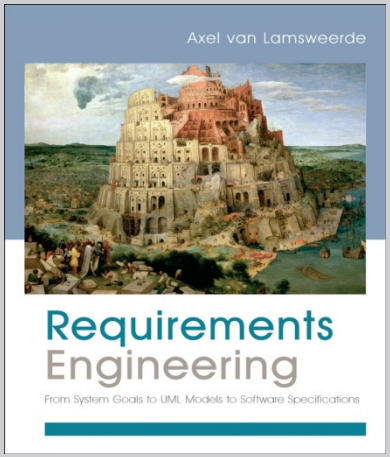
Requirements Engineering

From System Goals to UML Models to Software Specifications



Axel Van Lamsweerde



RE Temelleri

Bölüm 1

Hazırlık

Hazırlık Aşaması: özet

- RE nedir?
  - (Gerçek) dünya problemi ve problemin makine çözümü (world problem & machine solution)
  - RE: kapsamı Niçin(WHY), Ne (WHAT) ve Kim (WHO) boyutları
  - Kullanılan ifadeler: betimsel –tanımlayıcı (descriptive) ve kurallı (prescriptive)
  - Gereksinimlerin kategorileri: fonksiyonel ve fonksiyonel olmayan

Problem Dünyası -Problem World » ve «Makine Çözümü- Machine Solution »

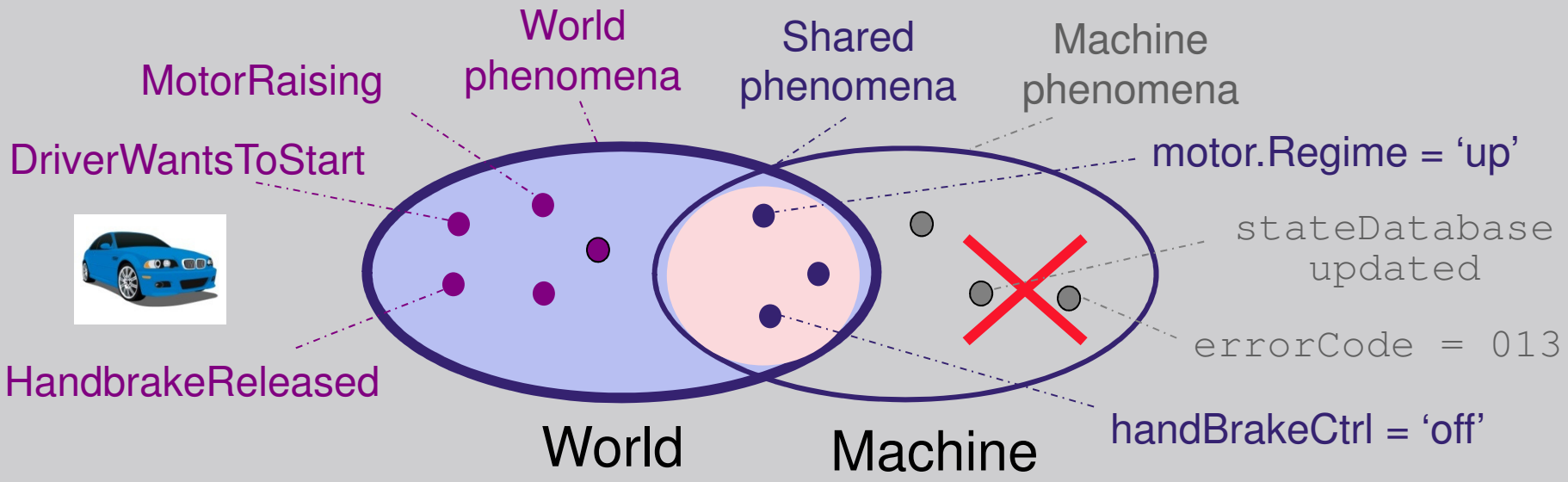
- Bir yazılımın çözümünden emin olmak için öncelikle gerçek dünya problemi doğru olarak çözülmelidir. Bu da problemi **tümüyle anlamak** ve **tanımlamaktır**.
  - Gerçek dünyada çözülecek problemin gereksinimleri ne (what)
  - Problemden ortaya çıkacak **içerik(** context)
- Örnek: araba kontrolü
  - Problem: el freninin indirilmesi bazı durumlarda istenilen şekilde gerçekleşemeyebilir
  - Context: araba sürme, fren, sürücünün amacı, güvenlik, kurallar…….

« Problem Dünyası – Problem World» ve «Makine Çözümü - Machine Solution »

- World: gerçek dünyanın çözülmesi istenen problem parçasıBunlar:
  - Beşeri Bileşenler (human components): organizasyon birimleri, personel(staff), operatorler, ...
  - Fiziksel Bileşenler: aygıtlar, eski yazılım, doğa ,.....
- Makine: problemi çözmek için düzenlenmesi gerekenler, yazılımın geliştirilmesi için gereksinimler nelerdir? Satın alınması gerekenler nelerdir?
  - Donanım/yazılımın impelementasyonu platformu , ortak giriş/çıkış aygıtları (sensorler & çalıştırıcılar (actuators)
- (RE) ilgi alanı...
  - istenilen makinenin **problem dünyasına etkisi**
  - problem dünyası ile ilgili **varsayımlar** (assumptions) ve uygun özellikleri (relevant properties)

Problem Dünyası ve Makine Çözümü

- Dünya ve makine sözcüklerinin kendi fenomenleri vardır.
- RE sadece **dünya** fenomeni (world phenomena) ile ilgilidir ve paylaşılanları da (shared phenomena) içerir.
  - Oysa yazılım tasarımı makine fenomeni ile ilgilidir.



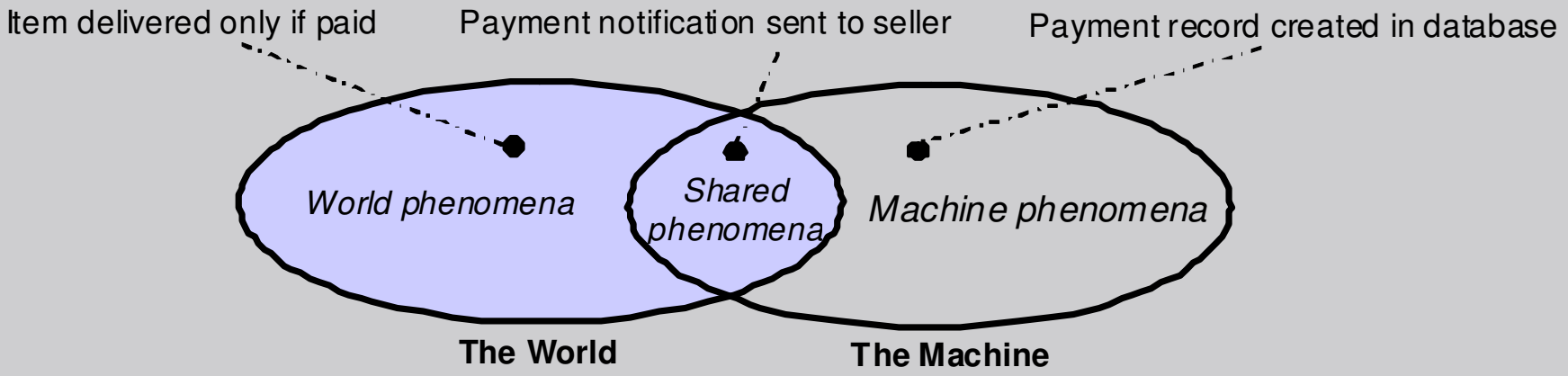
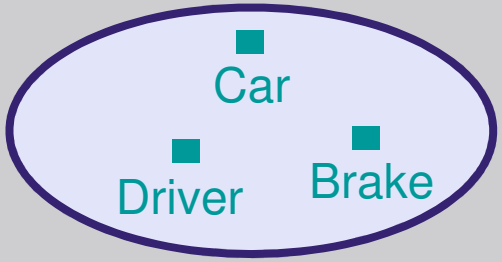


Figure 1.1 - The problem world and the machine solution

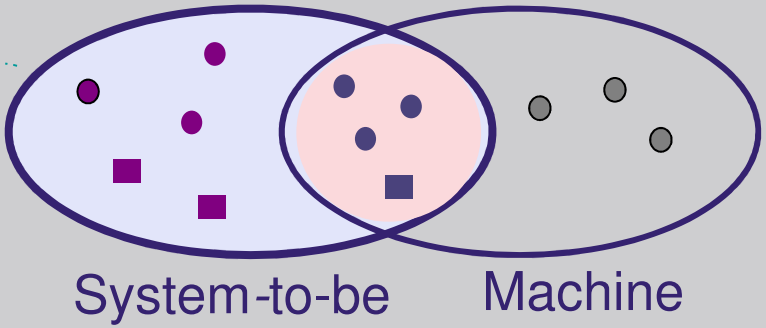
Problem dünyası (problem world) iki sistem versiyonu içerir

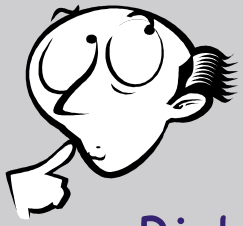
- System: problem dünyasında oluşturulmuş etkileşim halindeki bileşenlerin kümesi
- System-as-is: Makine oluşturulmadan önce sistemde bulunanlardır
- System-to-be: Makine sistem içerisinde çalıştırıldığında oluşacaklardır

Araba fren sistemi için kavramlar (concepts) , fenomemler (phenomena), kuralların betimlenmesi



kavramlar , fenomenler ,kurallar





RE: Tanımı

Birbiri ile bağlantılı, eşgünümlü) aktiviteler dizisi ...

- Yazılım yoğun bir sistem üzerinde hedefler, yetenekler, nitelikler, kısıtlamalar ve varsayımların (objectives, capabilities, qualities, constraints & assumptions) **keşfedilmesi, değerlendirilmesi, belgelenmesi, birleştirilmesi, düzeltilmesi ve uyarlanması** (exploring, evaluating, documenting, consolidating, revising and adapting)
- Problemlere bağlı olarak system-as-is şeklinde harekete geçen ve yeni teknolojilerle sağlanan fırsatlar (opportunities)

Diğer RE Tanımlamaları...

Ross'77

Gereksinimler tanımı aşağıdakileri işaret etmelidir: ...

- Mevcut ve gelecek (tahmini) koşullara göre niçin yeni bir sisteme ihtiyaç vardır ?
- **Hangi** (what) sistem özellikleri bu bağlamı sağlayacaktır ?
- Sistem nasıl (how) oluşturulacaktır?

Zave'97

- RE gerçek-dünyanın **amaçları, fonksiyonları,** yazılım sistemindeki **kısıtları** ile ilgilidir ve
  - Yazılım davranışının kesin betimlemeleri için bu kavramlarla bağlantı kurar
  - zaman içerisinde bu kavramların evrimleşmesi (evolution) sağlanır

Sistem gereksinimleri
Yazılım gereksinimleri

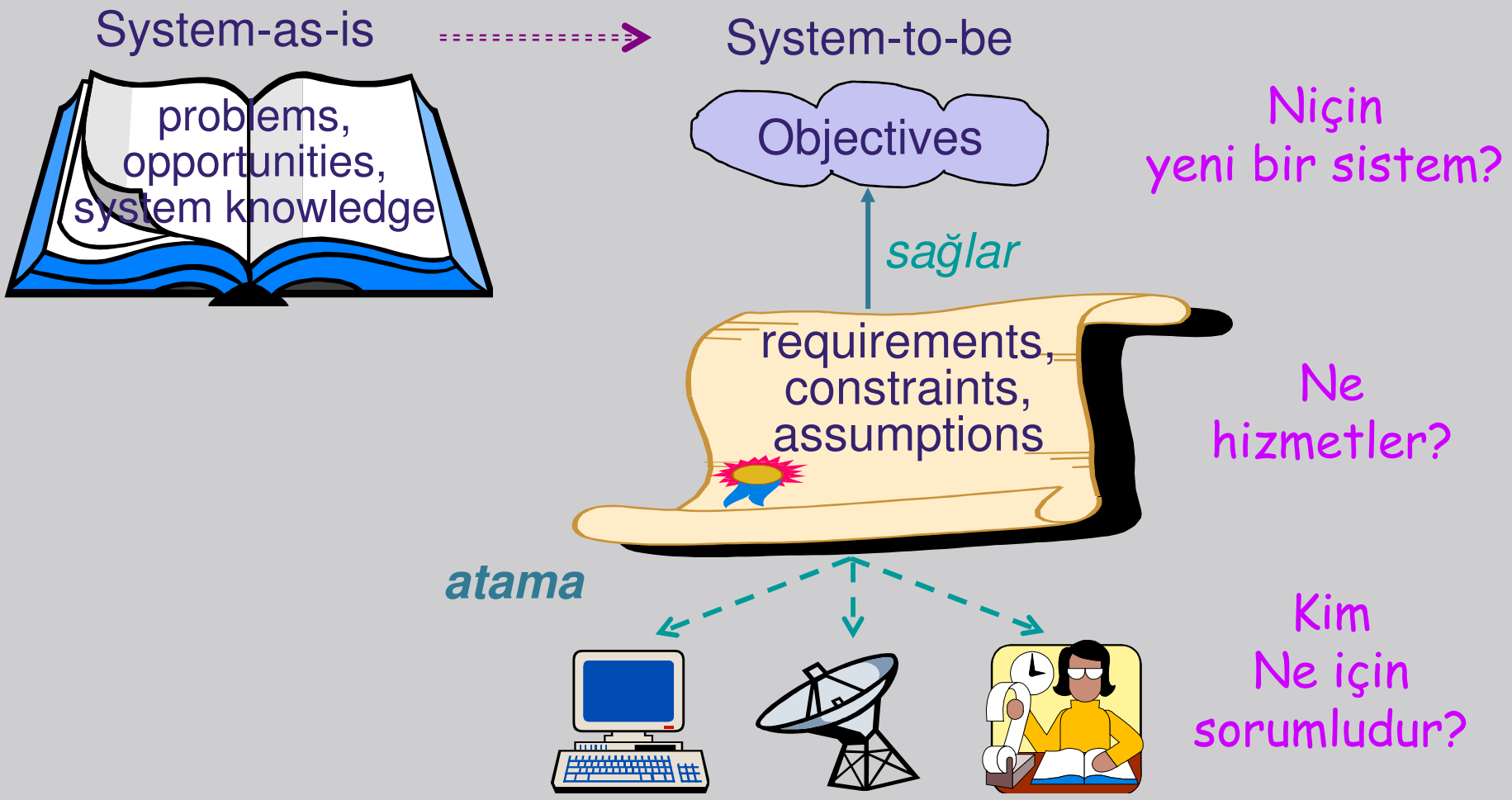
- Software-to-be: geliştirilecek yazılım makinenin parçasıdır, system-to-be 'nin bileşenidir.
- Environment: system-to-be 'nin diğer tüm bileşenleri insan, aygıtlar, önceden mevcut yazılım... içerir(people, devices, pre-existing etc.)
- System requirements: çevredeki (environments) fenomenler cinsinden formüle edilmiş olan **system-to-be** **ne** karşılamalıdır?

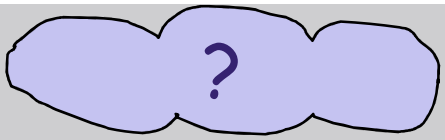
"El freni, sürücü hareket etmek istediğinde indirilmelidir.."

- Software requirements: : Yazılım ve çevresi tarafından paylaşılan (shared) fenomenler cinsinden formüle edilmiş olan **software-to-be** kendi başına **ne** karşılamalıdır?

"Yazılım çıktı değişkeni olan handBrakeCtrl 'ın değeri , yazılım girdi değişkeni motorRegime değeri arttığında kapalı olacaktır"

RE Kapsamı
Niçin, Ne ,Kim Boyutları
WHY, WHAT, WHO Dimensions



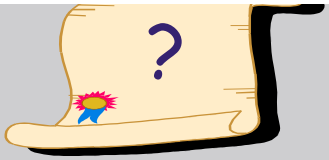


NİÇİN boyutu

- **system-to-be**'nin amaçlarının tanımlanması, analiz edilmesi ve gereksiz bilgilerden temizlenmesi (saflaştırılması)
  - **system-as-is** 'in analiz edilmiş eksikliklerinin belirlenmesi
  - ticari (iş-business) amaçları ile uyum sağlamak
  - Teknolojik fırsatlarından yararlanmak
- Örnek: havaalanı tren kontrol sistemi

  "Daha fazla yolcuya hizmet edebilmek"

  "Terminaller arasında transfer süresini azaltmak"
- Güçlükler
  - Alan bilgisinin edinilmesi (acquire domain knowledge)
  - Alternatif seçeneklerin değerlendirilmesi (yani aynı amacı sağlamak için alternatif yolların sağlanması)
  - Problemlerin –fırsatların eşleşmesi ve bunların değerlendirilmesi : etkileri, ilişkili riskler,
  - Çelişen amaçlarla başa çıkmak

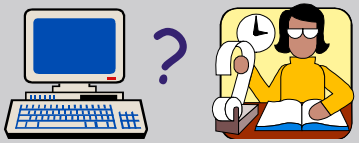


NE Boyutu

- system-to-be'nin **fonksiyonel servislerinin** tanımlanması (bunlar elle gerçekleştirilen işlemlerle bağlantılı yazılım servisleridir)
  - Tanımlanmış amaçları sağlamak için to satisfy the identified objectives
  - Nitelik kısıtlarına göre: güvenlik ,performans,.....
  - Çevre ile ilgili gerçekçi varsayımlara dayanan
- Örnek: havaalanı tren kontrolü

  "trenin güvenli hızlanmasının hesaplanması"

  "trendeki yolculara yararlı bilgilerin gösterilmesi"
- Güçlükler
  - Özellikler dizisinin doğru tanımlanması
  - Bunların tüm kısımlar tarafından kesin olarak anlaşılmasının sağlanması
  - Sistem amaçlarının geriye doğru izlenebilirliğinin sağlanması



KİM Boyutu

- **System-to-be** bileşenleri arasında amaçlar, servisler, kısıtları için sorumlulukların belirlenmesi
  - Yazılım-ortamı sınırlarını oluşturmak üzere
- Örnek: havaalanı tren kontrolü
  - "Trenin güvenli hızlanması" ... software-to-be 'nin doğrudan sorumluluğu altında (sürücüsüz seçenek) *veya* sürücünün yazılım işaretlerini takip etmesi
  - "Treni hız/pozisyonunun tahmini doğruluğu»… ... İzleyen sistemin sorumluluğunda ***ya da*** önceki trenin sorumluluğunda ?
- Güçlükler
  - Doğru otomasyon derecesine karar vermek için alternatif seçeneklerin değerlendirilmesi