

Yazılım Gereksinimleri Mühendisliği

Temel Bilgiler

1. Hafta

26 Şubat 2026

❑ **Requirements Engineering Magazine** <https://re-magazine.ireb.or>

Derslerin yürütümü süresince ilgili makaleler buradan okunacaktır.

İlk hafta dersinin kaynak makalesi :

<https://re-magazine.ireb.org/print/requirements-elicitation-in-modern-product-discovery>

❑ Bazı haftalar dersin içeriği ile ilgili **video linkleri** paylaşılacaktır.

İlk hafta dersinin kaynak videosu: <https://www.youtube.com/watch?v=qENBiYaAXNE>

Okuma parçaları ve video linkleri doğrudan ilgili hafta işlenecek konularla ilgilidir.

❑ İşlenecek dersler haftalık olarak dersin izlencesinden takip edilebilir.

❑ **Dersin Değerlendirmesi** : Ara Sınav %25

Derste yapılacak ortak vaka çalışmaları ve Proje Çalışması %35

Final %40

Proje çalışması yarıyıl süresince devam edecek bir süreç olup, sadece sisteme yüklenmesi yeterli değil derste elden teslimi zorunludur.

❑ **Temel ders kitabı**: Axel von Lamsweerde, Requirements Engineering From System Goals to UML Models to Software Specification



Resmin Açıklaması

Pusula: Geleneksel gereksinim mühendisliği, nereye gidileceğini bilmektir.

Turbo Motor: Yapay Zeka ile hızlıca hedefe ulaşılmasını belirtir.

Resmin Verdiği Mesaj: Pusula yanlış yöne işaret ediyorsa gereksinimlerin belirlenmesinde motorun hiçbir faydası olmayacaktır.

Yapay Zeka tarafından oluşturulan bu resim Gereksinim Mühendisliğini ve Yapay Zekanın rolünü ifade etmektedir.

Agile Requirements Engineering

Kullanıcı Hikayelerini Haritalama: Büyük özellikleri yönetilebilir parçalara ayrılmasının öğrenilmesidir.

Ürün Geri Bildirimi (backlog) Düzenleme: Yapılan işin değeri (business value) ve teknik tartışmalara göre gereksinimleri önceliklendirmenin anlaşılmasıdır.

Kabul Kriterleri: Davranış Odaklı Geliştirme (Behaviour Driven Development -BDD) gibi çerçeveler kullanarak test edilebilecek koşulların yazılmasıdır.



Gereksinimlerin Prototiplerinin Oluřturulması

Kategori	Modern Yaklařım	Kullanılan Araçlar (Tools)
Elicitation (Bilgi toplama)	Role-playing and Prototyping	Figma, Miro, Balsamiq
Documentation	Living Documentation & Wikis	Confluence, Notion, Markdown
Management (yönetim)	Issue Tracking & Traceability (Sorunların takibi ve izlenebilirlik)	Jira, Trello, Azure DevOps
Modeling (modelleme)	Lean UML and SysML	Lucidchart, Draw.io, Enterprise Architect



- ❑ Bu ders, yazılım gereksinimlerinin belirlenmesi, analizi, spesifikasyonu ve yönetimi konularını kapsamaktadır.
- ❑ Geleneksel modelleme tekniklerinde uzmanlaşmak hedef iken, aynı zamanda **büyük dil modellerini** (Large Language Models- LLMs) kullanarak dokümantasyonun otomatikleştirilmesi, paydaşların (**stakeholders**) simüle edilmesi ve karmaşık sistem mantığının doğrulanması (**validation**) öğrenilir.

Yazılım-Yoğun Sistemlerin Geliştirilmesi

- ❑ Herhangi bir yazılım ürününün tasarımının gerçekleştirilmesi için **işlevsel ve fiziksel** ihtiyaçlarının belirlenmesi gerekir.
- ❑ Bu fiziksel ve işlevsel ihtiyaçlar, ürünün "**gereksinimlerine**" katkı sağlar.

- ❖ **İşlevsel ihtiyaçlar** fonksiyonel gereksinimleri,
- ❖ **Fiziksel ihtiyaçlar** fonksiyonel olmayan gereksinimleri tanımlar.

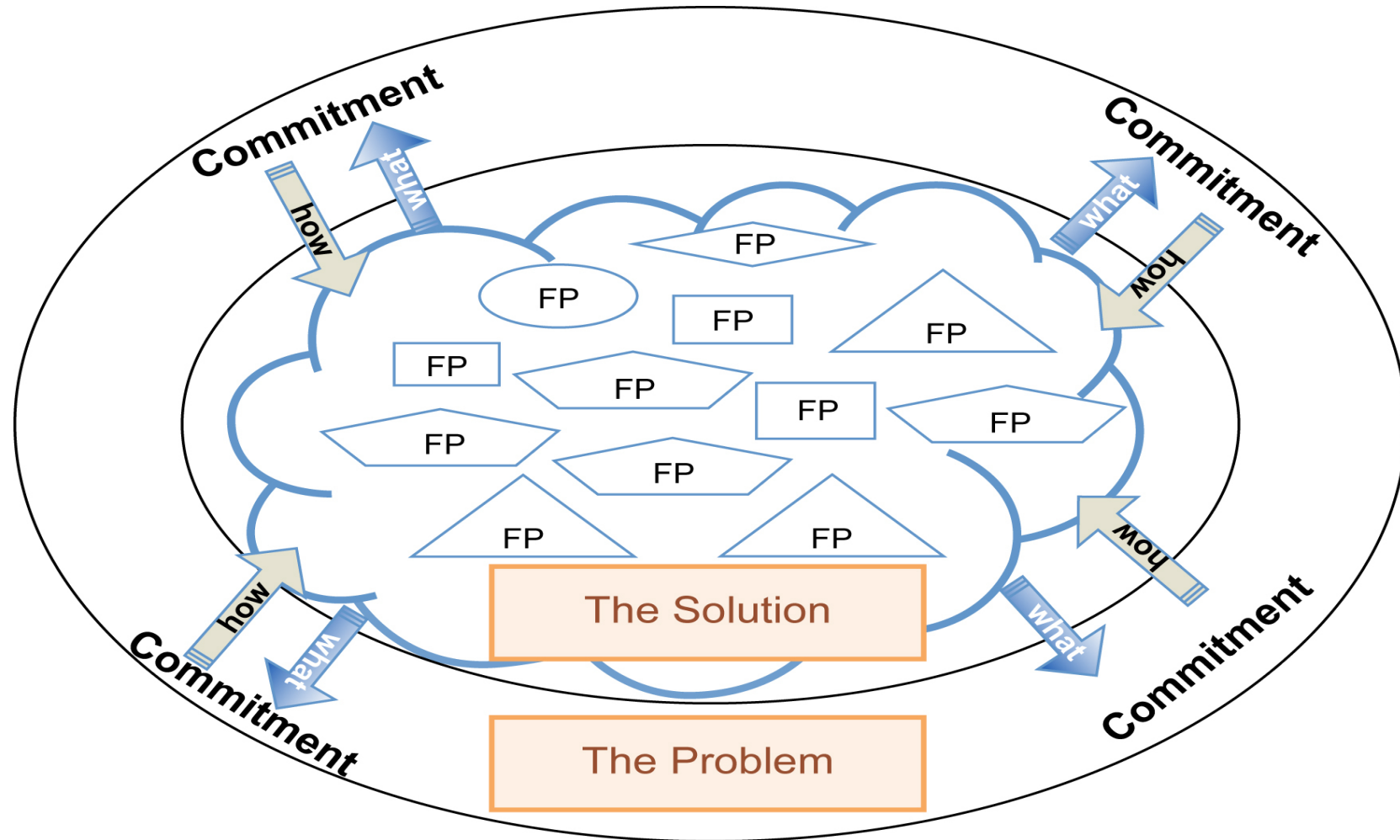
- ❑ Yazılım ve yazılım yoğun sistemler de istenilen yazılım ürünleri oluşturur.

IEEE Requirements Standard

Institute of Electrical and Electronics Engineers standardı 610.12-1990 (IEEE-Std-610.12-1990) gereksinimler sözcüğünü şöyle tanımlar:

1. Kullanıcının bir sorunu çözmek veya bir amaca ulaşmak için ihtiyaç duyduğu herhangi bir koşul veya yapabileceği şeyler.
2. Bir sözleşmeyi, standardı, şartnameyi veya diğer resmi belgeleri sağlamak üzere bir sistem veya sistem bileşeni tarafından sağlanması gereken koşul veya yapabilecekler.
3. Bir koşulun veya yeteneğin 1 veya 2'dekileri sağlayacak şekilde belgelenmesi.

Gereksinimler Arasındaki İlişkiler What ve How Sorusu

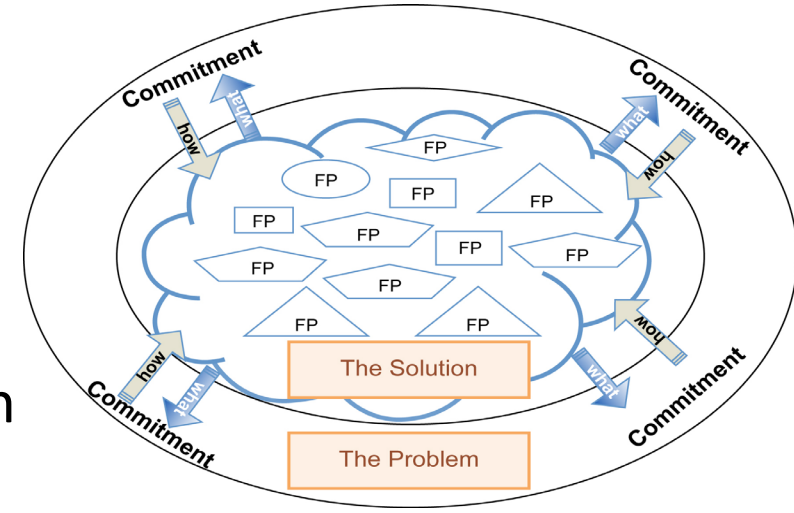


Sistem Gereksinimleri

❑ Sistem güvenilir olmalıdır.

❖ Açık bir ağ ortamında çalışırken bile, kendisini potansiyel kötü niyetli saldırılardan koruyabilmelidir.

❖ Çeşitli virüs saldırılarını algılayıp önleyebilmeli ve sistemin anahtar bileşenlere virüs bulaştığında acil durum sürecini başlatabilmelidir.



❑ Sistem kullanıcı dostu bir arayüz sunmalıdır.

❖ Sistem seviyesi gereksinimlerdir.

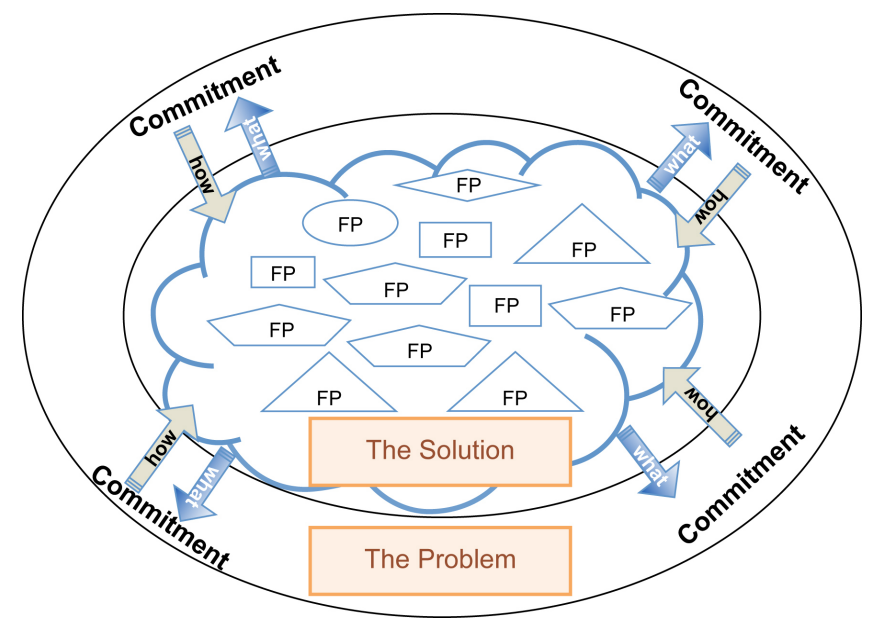
❑ Kullanıcının tuş takımına girdiği kişisel kimlik numarası doğruysa sistem kapıyı açmalıdır.

❖ Fonksiyon noktası (FP) düzeyindeki gereksinimlerdir.

❑ Sistem, tarih, saat ve kişinin kimliği gibi her erişimi kaydetmelidir.

❖ Fonksiyon noktası (FP) düzeyindeki gereksinimlerdir.

System –to- be nedir?



❑ Geliştirilen üründür.

❖ Ürünün yükümlülüğü geliştirilecek sistemin sahip olacağı fonksiyon noktalarıdır.

❑ *what-how çifti*: Gereksinmeler ile «*ne geliştirilecektir?*» sorusu belirlenirken, sistem ise «*sistem nasıl geliştirilecektir?*» sorusunu araştırır.

❑ Gereksinimler mühendisliği hem «*how*», hem de «*what*» ile ilgilidir ve ikisi arasında köprü görevini üstlenir.

Sonuç olarak:

Gereksinimler, ürün ve yazılım ile ilgili tüm ifadeleri ve iddiaları (assertions) ve aradaki fenomeni (phenomena) içerir.

Problem Definition: What

R1.6 The navigation system shall allow the driver to enter the destination of the trip conveniently.

Solution Definition: How

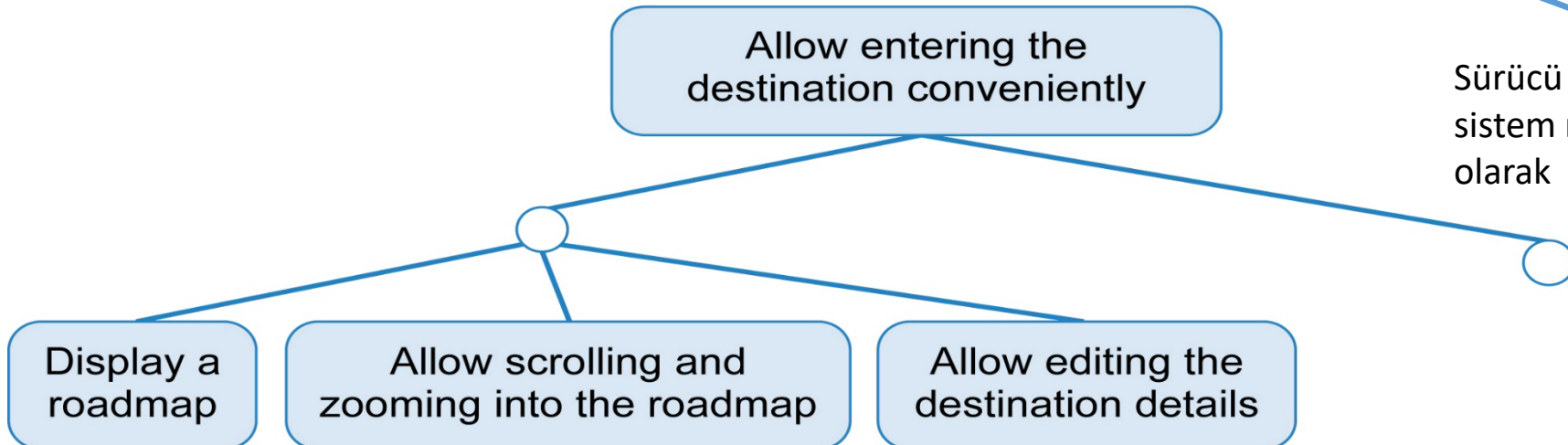
R1.7 When the driver starts a new trip, the navigation system shall display a roadmap of the area centered on the current position.

R1.8 The navigation system shall allow the driver to scroll and zoom into the roadmap.

R1.9 After the driver has selected a destination on the roadmap, the system shall allow the driver to edit the destination details, e.g. city, street, etc.

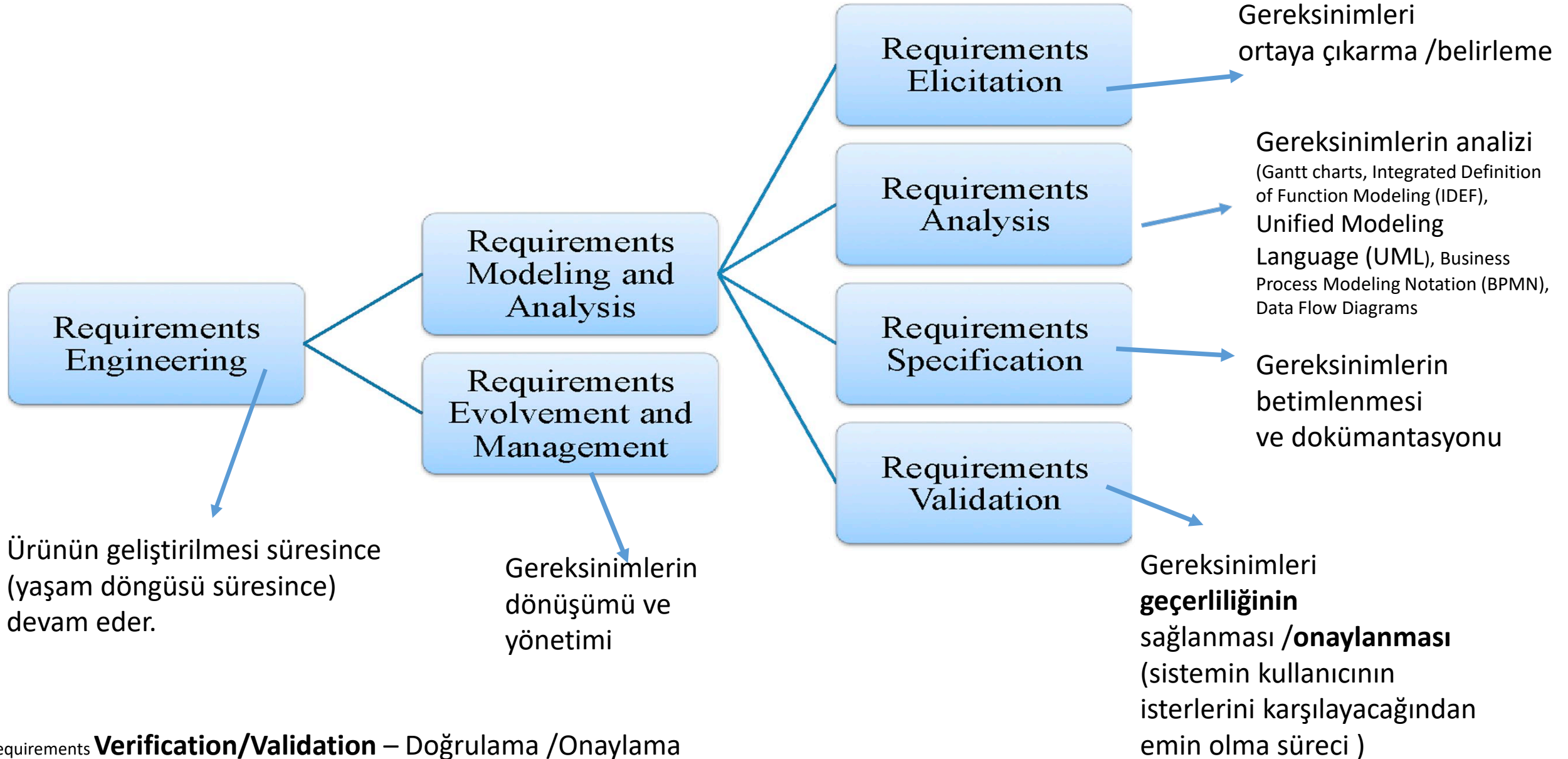
Navigasyon sistemi, sürücünün gideceği yeri girmesine izin verecektir. «Shall» cümlesi

Excerpted from (Pohl, 2010)



Sürücü yeni bir rota belirlediğinde, sistem mevcut pozisyonu ortalanmış olarak bir yol haritası çizer.

Gereksinimler Mühendisliği



Bir terim olarak Gereksinim Mühendisliği

"Gereksinim mühendisliği" terimi,

- i) çözülmesi amaçlanan (purpose) problem,
- ii) amacı gerçekleştirmek için yapılabilecekler (ability) ,
- iii) «**system to- be**» yani sistemi geliştirmek üzere hedeflenen eylemler (yapılabilecekler)
- iv) sistem sorununu çözecek **kısıtların (constraints)**

bir koleksiyonu olarak ifade edilir.

Kısaca:

Amacın belirlenmesi, iletilmesi ve belgelenmesi ile ilgili bir dizi faaliyet anlamına gelir.

system-as-is ve *system to-be*

- ❑ Gereksinim mühendisliği süreci, mevcuta göre daha yenilikçi bir uygulama talebi olduğunda veya «system as- is» sıkıntıda olduğunda başlatılır.
- ❑ Gereksinim mühendisliği uygulandığında talepler karşılanır veya eski sistemi değiştirmek için kullanılabilecek yeni bir sistem spesifikasyonunun üretimi amaçlanır.
- ❑ Gereksinim mühendisliği, kullanıcıların, müşterilerin ve sistemden etkilenen diğer bileşenlerin **gerçek dünyadaki ihtiyaçları (amaç ve kısıtlar)** ile bilgi teknolojilerinin sağlayabileceği yetenekler ve fırsatlar arasındaki bir köprüdür.

Gereksinimler Mühendisliđi nereden dođdu?

1965 – 1985 arası: Yazılım Krizi

- ❑ Yazılım krizi terim olarak ilk defa 1968 yılında Almanya'da yapılan NATO Software Engineering konferansında F.L. Bauer tarafından kullanılmıştır.
«*Mevcut Sistem tasarımı yaklaşımı fazlasıyla ampiriktir.*
Artan sistem karmaşıklığı ile baş edememektedir».
- ❑ 1960'larda, 1970'lerde ve 1980'lerde yazılım geliştirmede bir takım sorunlar tespit edildi.
- ❑ Yazılım projelerinin karşılaştığı sorunlar şunlardı:
 - ❖ Projeler bütçesini aşıyordu,
 - ❖ Proje kullanılan özelliklerine, bunlardan çalışan olanlarına bile zarar veriyordu.



1965 – 1985 arası: Yazılım Krizi

□ Maliyet ve Bütçe Aşmaları: OS/360 işletim sistemi klasik bir örnekti.

❖ 1960'lı yıllara ait on yıllık projeydi

✓ Proje sonunda zamanın en karmaşık yazılım sistemlerinden biri olarak üretildi.

❖ OS/360, ilk büyük (1000 programcı) yazılım projelerinden biriydi.

❖ Fred Brooks, *The Mythical Man Month* (1975) adlı kitabında, geliştirmeye başlamadan önce tutarlı bir mimari geliştirmemek gibi milyonlarca dolarlık bir hata yaptığını iddia ediyor.

□ Özellik Hasarı: Yazılım kusurları (defects) maddi hasara neden olabilir.

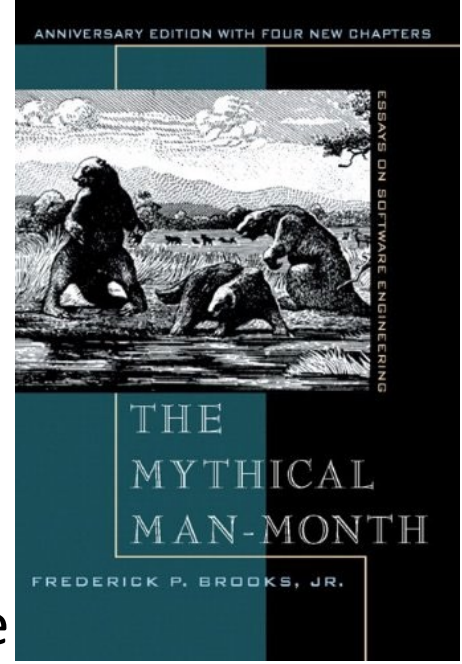
❖ Kötü yazılım güvenliği, bilgisayar korsanlarının kimlikleri çalmasına, zaman, para ve itibar kaybına neden olacaktır.

□ Yaşam ve Ölüm: Yazılım kusurlarının (defects) öldürme gücü vardır.

❖ Radyoterapi makinelerinde kullanılan bazı gömülü sistemler son derece yıkıcı şekilde başarısız oldu.

✓ Hastalara ölümcül dozlarda radyasyon verildi.

✓ Bu başarısızlık vakasının en çok adı geçeni *Therac 25* olayıdır.



Yazılım Krizindeki Sorumluluğun Nedenleri -I

□ Yazılım krizinin nedeni, geliştirme süreciyle ilgili tüm sorun ve karmaşıklıklarla bağlantılıdır.

Yazılım krizine katkısı olan çeşitli faktörlerden birkaçı aşağıda açıklanmaktadır.

□ *Ölçeklenebilirlik sorunu*: Yüz binlerce satıra küçük bir yazılım geliştirmenin yolları.

❖ Başka bir deyişle, küçük sistemler geliştirmek için kullanılan yöntemler genellikle büyük sistemlere ölçeklenmez

□ *Yazılım pahalıdır*: Yazılım geliştirmenin maliyeti donanıma göre yüksektir.

❖ Yazılımı geliştirmek için yüksek nitelikli ve kalifiye insan gücüne ihtiyaç vardır

❖ Masrafların artmasına sebep olacak şekilde büyük miktarda para ödemeleri gerekir.

□ *Yazılım gecikir* : «Geç» terim olarak, yazılımın belirtilen süre içinde tamamlanamayacağı anlamına gelir.

❖ Yazılım zamanında teslim edilmez.

□ *Yazılıma güvenilmez*: Güvenilmezlik (unrealibility) , yazılımın gerekli işlevi yerine getirmediği anlamına gelir.

❖ Yazılıma giren hatalar (*bugs*) nedeniyle birçok sorunla karşılaşılır (*failure*).

Yazılım Krizindeki Sorumluluğun Nedenleri -II

❑ *Üretkenlikteki tutarsızlık (inconsistency)*

- ❖ Programcıların üretkenliği talebi karşılayamaz.
- ❖ Yazılım geliştirme süresi ve maliyeti ile ilgili oldukça iyimser tahminler yapılır.
- ❖ Mevcut sorun ve sorunun çevresi iyi anlaşılabilir.
- ❖ Geliştirilen yazılım, sistemin kullanım amacına pek uygun değildir.

❑ *Kodun bakımındaki (maintenance) zorluk*

- ❖ Bakım, mevcut yazılımı anlamak etrafında yoğunlaşır ve bakım yapanlar, zamanlarının çoğunu değiştirmek istedikleri yazılımı anlamak için harcamak zorunda kalırlar.

❑ *Problem alanının karmaşıklığındaki artış*

- ❖ Büyük ve daha karmaşık yazılım sistemlerine olan talep artmaktadır.

❑ *Çabaların (efforts) arttırılması*

- ❖ Yazılım geliştirme faaliyetlerinin çoğunda otomasyon olmaması nedeniyle çalışmaların tekrarlanması gerekir.

Devam Etmekte olan Yazılım Krizinin Nedenlerinin Özeti

- Projeler bütçeyi aşar.
- Projeler zamanla yarışır.
- Yazılım çok verimsizdir.
- Yazılım düşük kalitede üretilmiştir.
- Yazılım genellikle gereksinimleri karşılamaz.
- Projeler yönetilememektedir ve kodun bakımı zordur.
- Yazılım hiç bir zaman teslim edilemez.

1985 – 1989 arası : Gümüş Mermi (Silver Bullet)

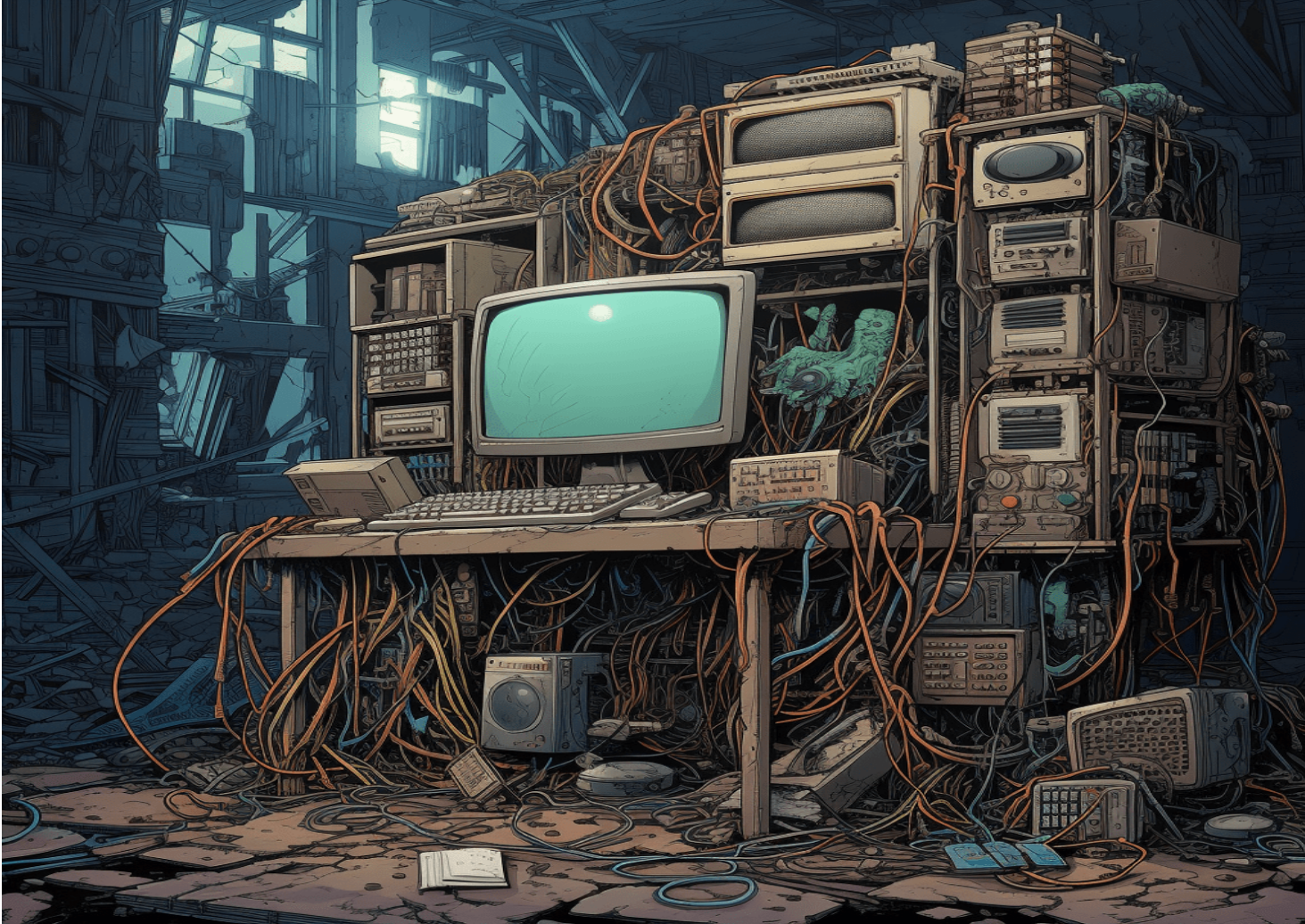
- ❑ Yazılım krizini çözmek, çok uzun zamandır araştırmacıları ve yazılım araçları üreten şirketleri çok fazla zorlamaktaydı..
 - ❑ Araştırmacılar, yazılım krizini çözmek için 1970'lerden 1990'lara kadar her yeni teknolojiyi ve uygulamayı "gümüş mermi" olarak denediler.
- Sonunda aşağıdakileri gümüş mermi olarak onayladılar:
- ❑ **Araçlar:** Araçlar özellikle vurgulandı: yapılandırılmış programlama, nesne yönelimli programlama, CASE araçları, Ada, belgeler ve standartlar gümüş mermiler olarak tanıtıldı.
 - ❑ **Disiplin:** Bazı uzmanlar, yazılım krizinin programcıların disiplin eksikliğinden kaynaklandığını savundu
 - ❑ **Biçimsel yöntemler:** Bazı araştırmacılar, yazılım geliştirmeye formel mühendislik metodolojileri uygulanırsa, yazılım üretiminin diğer mühendislik dalları kadar öngörülebilir bir endüstri olacağına inanıyordu.
 - ❑ **Süreç:** Birçok profesyonel, Yetenek Olgunluk Modeli (CMM) gibi tanımlanmış süreçlerin ve metodolojilerin kullanımını savundu.
 - ❑ **Profesyonellik:** Bu öge, etik kurallar, lisanslar ve profesyonellik üzerinde çalışmalara neden oldu.



ARIANE 5 First Flight Failure - Flight 501, June 4, 1996

<https://share.google/kXzrtjVTWSGrLxrD1>

The National Programme for IT (NPfIT) failed



2002'de başlatılan ve 2011'de iptal edilen bu proje, İngiliz vergi mükelleflerine yaklaşık 10 milyar sterlin (12,7 milyar dolar) maliyetle tarihin en pahalı başarısız sivil bilişim projelerinden biri olma derecesini sürdürüyor.