

Yazılım Mühendisliği

1906003082015

Dr. Öğr. Üy. Önder EYECİOĞLU
Bilgisayar Mühendisliği



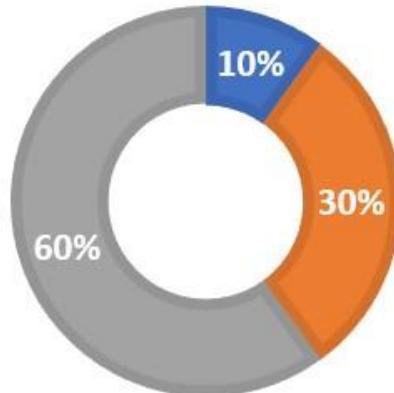
Giriş

Ders Günü ve Saati:

Salı: 09:15-13:00

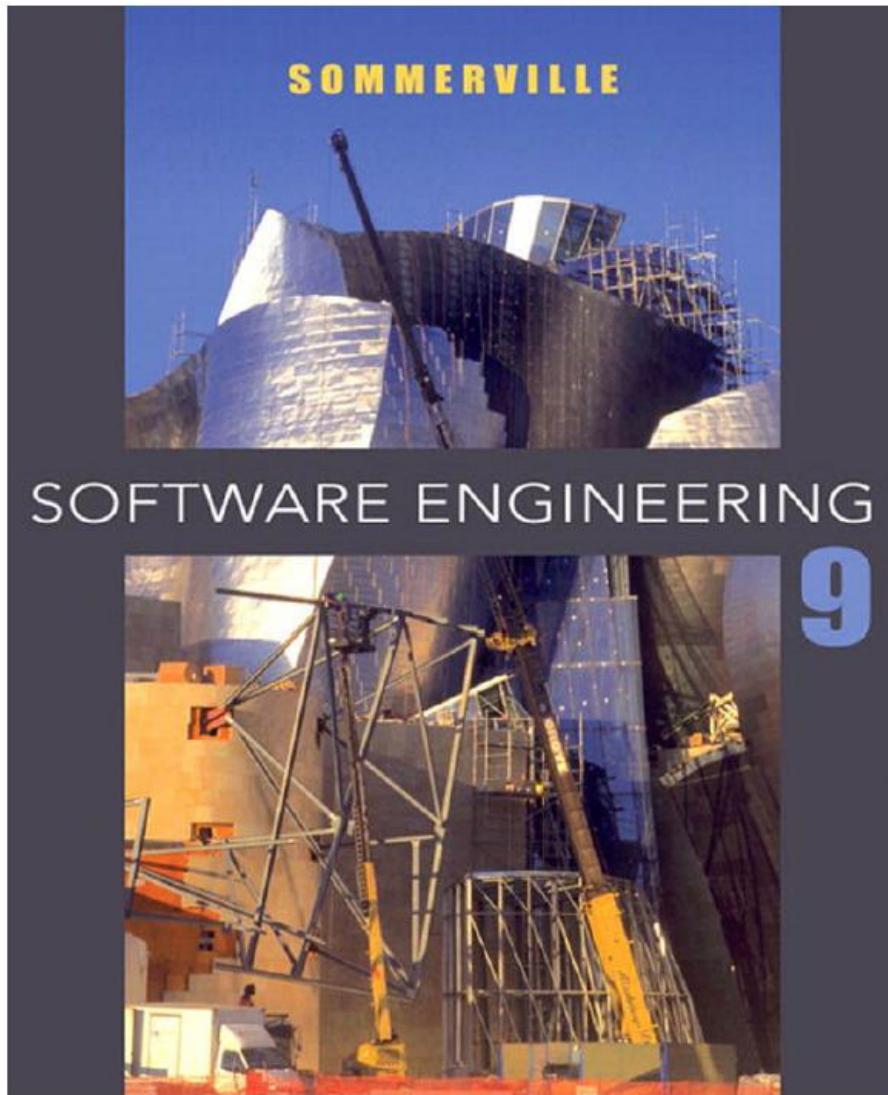
Devam zorunluluğu %70

■ Ödev ■ Vize ■ Final



HAFTA	KONULAR
Hafta 1	Yazılım Mühendisliğine Giriş
Hafta 2	Yazılım Geliştirme Süreç Modelleri
Hafta 3	Yazılım Gereksinim Mühendisliği
Hafta 4	Yazılım Mimarisi
Hafta 5	Nesneye Yönelik Analiz ve Tasarım
Hafta 6	Laboratuar Çalışması: UML Modelleme Araçları
Hafta 7	Yazılım Test Teknikleri
Hafta 8	Ara Sınav
Hafta 9	Yazılım Kalite Yönetimi
Hafta 10	Yazılım Bakımı - Yeniden Kullanımı ve Konfigürasyon Yönetimi
Hafta 11	Yazılım Proje Yönetimi (Yazılım Ölçümü ve Yazılım Proje Maliyet Tahmin Yöntemleri)
Hafta 12	Yazılım Proje Yönetimi (Yazılım Risk Yönetimi)
Hafta 13	Çevik Yazılım Geliştirme Süreç Modelleri
Hafta 14	Yazılım Süreci İyileştirme, Yeterlilik Modeli (CMM)

Kaynaklar



3.Hafta
GEREKSİNİM MÜHENDİSLİĞİ



Kullanıcı gereksinimleri ve sistem gereksinimleri aşağıdaki gibi tanımlanabilir:

1. Kullanıcı gereksinimleri sistemin kullanıcıya ne tür servisler sunacağı ve hangi kısıtlamalar altında çalışmak zorunda olduğunun doğal dilde ve çizimler ile ifadesidir. Kullanıcı gereksinimleri gerekli sistem özelliklerinin genel olarak anlatıldığı ifadelerden sistem işlevselliliğinin ayrıntılı ve kesin olarak tariflendiği ifadelere kadar değişebilir.
2. Sistem gereksinimleri yazılım sisteminin fonksiyonlarının, servislerinin, işletme kısıtlarının çok daha ayrıntılı açıklamasıdır. Sistem gereksinimleri dokümanı (bazen fonksiyonel spesifikasyon olarak adlandırılır) tam olarak ne gerçekleştirileceğini tanımlamalıdır. Sistemi satın alan ve yazılım geliştiriciler arasındaki kontratın bir parçası olabilir.



Kullanıcı gereksinimleri tanımı

1. ZihinSaS sistemi her ay her klinik için, o ay, o klinikte reçetelenen ilaçların maliyetini gösteren aylık yönetim raporları üretmelidir.

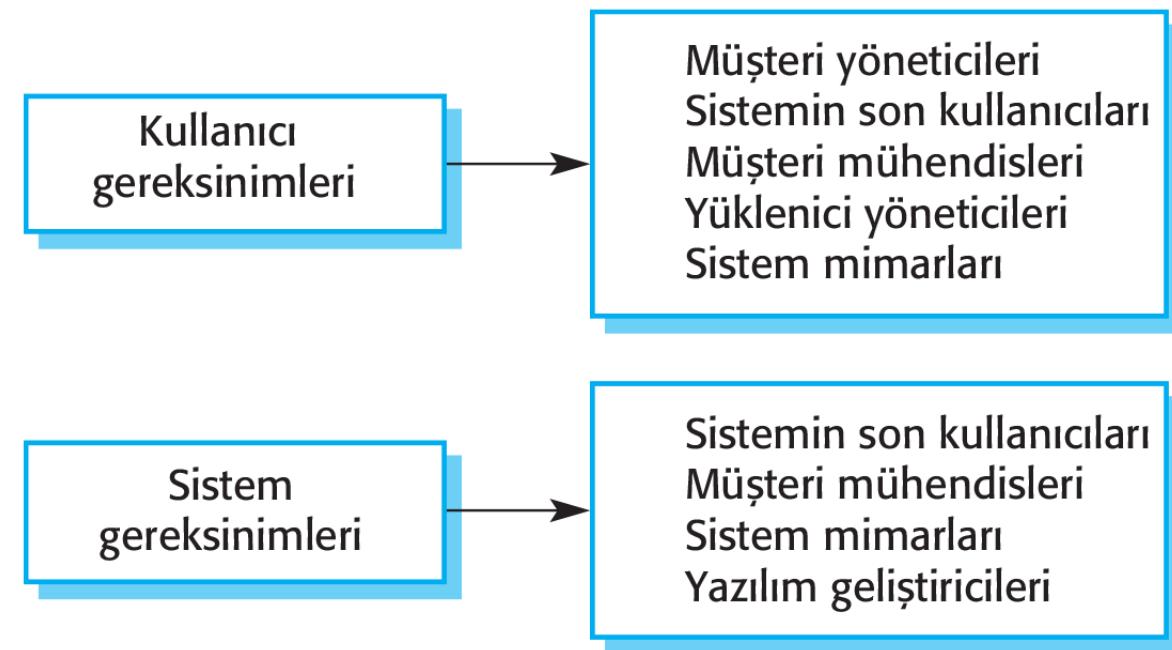
Sistem gereksinimleri tanımı

- 1.1 Her ayın son çalışma gününde, reçetelenen ilaçların, maliyetlerinin, hangi klinik tarafından reçetelelendiklerinin gösterildiği özet rapor hazırlanmalıdır.
- 1.2 Sistem raporu basılması için ayın son çalışma gününde saat 17.30'dan sonra hazırlamalıdır.
- 1.3 Rapor her klinik için hazırlanmalıdır ve tek tek ilaç isimlerini, toplam reçete sayısını, reçetelenen doz sayısını ve reçetelenen ilaçların toplam maliyetini listelemelidir.
- 1.4 Eğer ilaçlar farklı dozlarda bulunabiliyorsa (10 mg'luk, 20 mg'luk gibi) her doz birimi için ayrı rapor hazırlanmalıdır.
- 1.5 İlaç maliyet raporlarına erişim yönetim erişim kontrol listesinde gösterilen yetkili kullanıcılar ile sınırlı olmalıdır.

Şekil 4.1 Kullanıcı ve sistem gereksinimleri



Şekil 4.2 Farklı
gereksinim
tanımlamalarının
okuyucuları



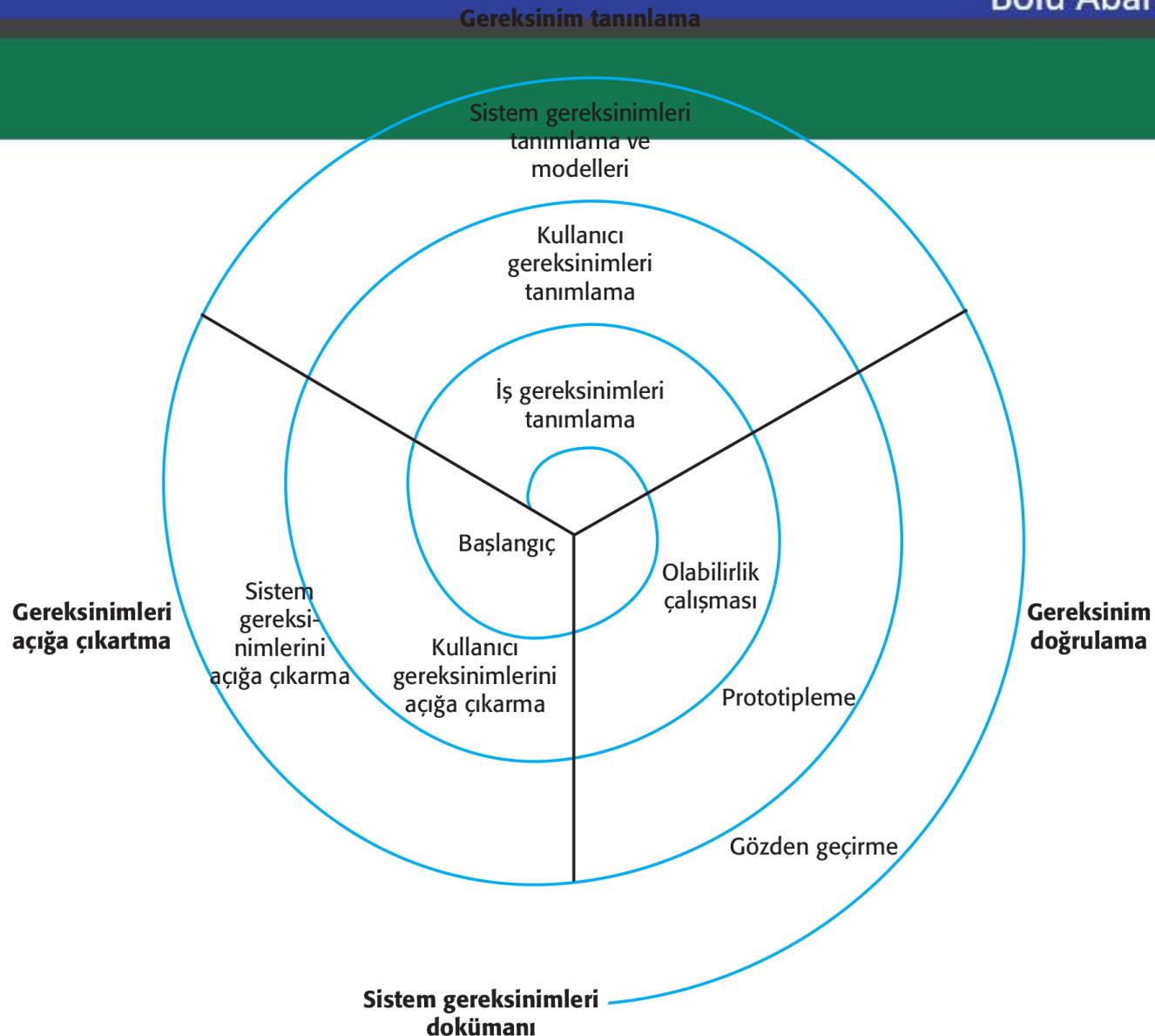
GEREKSİNİM MÜHENDİSLİĞİ SÜRECİ

İkinci bölümde tartıştığım gibi, gereksinim mühendisliği üç temel eylem içerir. Bunlar, paydaşlar ile etkileşerek gereksinimleri keşfetmek (açığa çıkartmak ve analiz); bu gereksinimleri standart biçimde çevirmek (tanımlama); ve gereksinimlerin gerçekten kullanıcının istediği sistemi tanımladıklarını kontrol etmektir (doğrulamak).

Şekil 2.4'te bunları ardışık süreçler olarak göstermiştim. Bununla birlikte, pratikte gereksinim mühendisliği eylemlerin dönüşümlü çalıştığı yinelemeli bir süreçtir. Şekil 4.6 bu dönüşümü gösterir.

Eylemler bir sarmal etrafında yinelemeli bir süreç olarak düzenlenmiştir. GM sürecinin çıktısı sistem gereksinimleri dokümanıdır. Bir döngüde her eylem için harcanan zaman ve emek tüm sürecin evresine, geliştirilen sistemin tipine ve ayrılmış bütçeye bağlıdır.





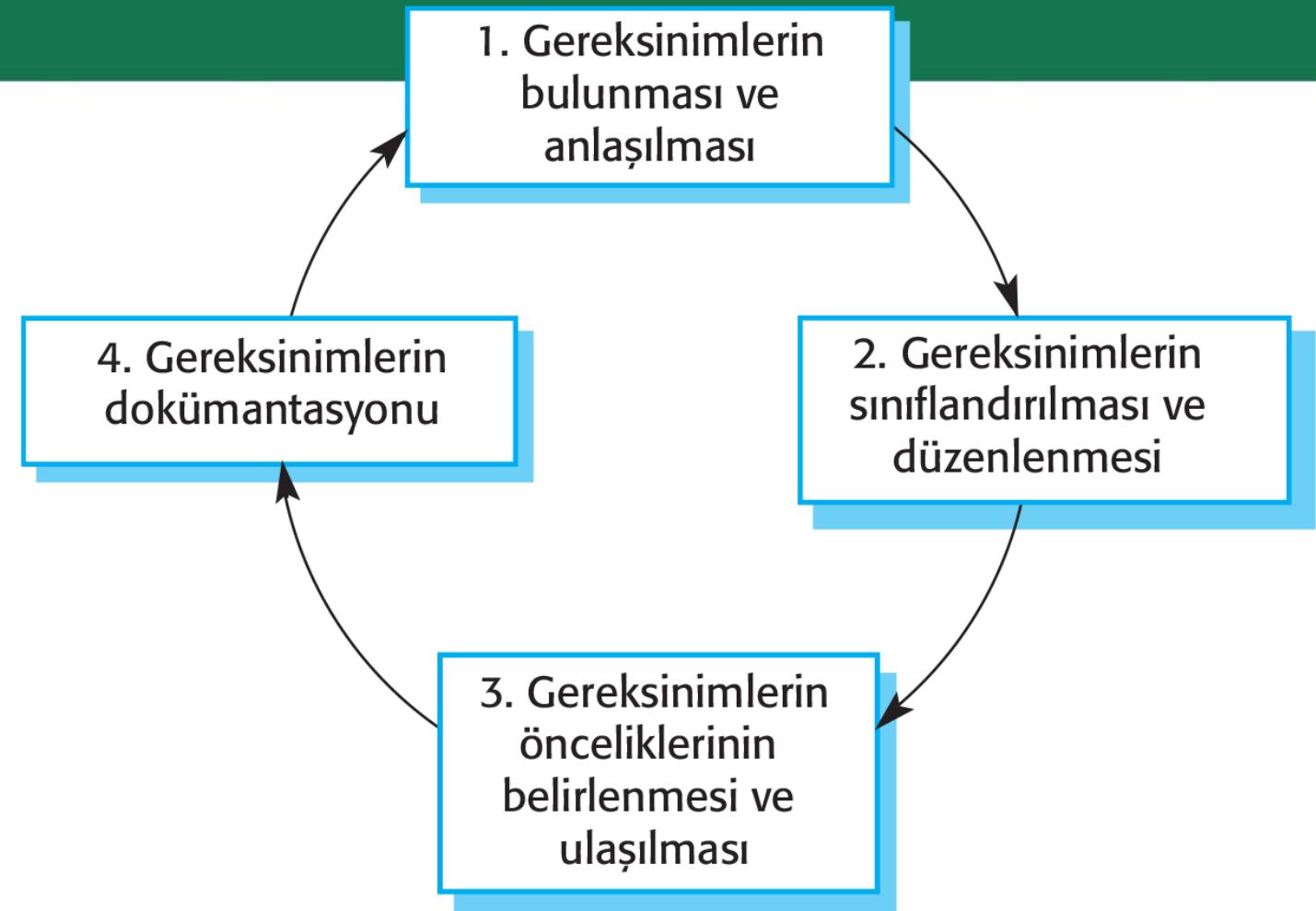
Şekil 4.6 Gereksinim mühendisliği sürecinin sarmal görüntüsü

GEREKSİNİMLERİN AÇIĞA ÇIKARILMASI

Gereksinimlerin açığa çıkarılması sürecinin amaçları paydaşların yaptıkları işin ve yeni sistemi bu işi desteklemek için nasıl kullanabileceklerinin anlaşılmasıdır. Paydaşlardan gereksinimleri açığa çıkarmak ve anlamak birkaç nedenden ötürü zor bir süreçtir:

1. Paydaşlar genellikle çok genel gereksinimler dışında bir bilgisayar sisteminden ne istediklerini bilmezler; sistemin ne yapmasını istediklerini dile getirmekte zorlanabilirler; yapılabılır ve yapılamaz şeyleri bilmedikleri için gerçekçi olmayan isteklerde bulunabilirler.
2. Bir sistemdeki paydaşlar doğal olarak gereksinimleri kendi terimleri ve işlerinin üstü kapalı bilgisi ile ifade eder.
3. Değişik gereksinimlere sahip farklı paydaşlar gereksinimlerini farklı yollardan ifade edebilir.
4. Politik etkenler bir sistemin gereksinimlerini etkileyebilir. Yöneticiler kuruluşa etkilerini arttıracak sistem gereksinimleri talep edebilir.





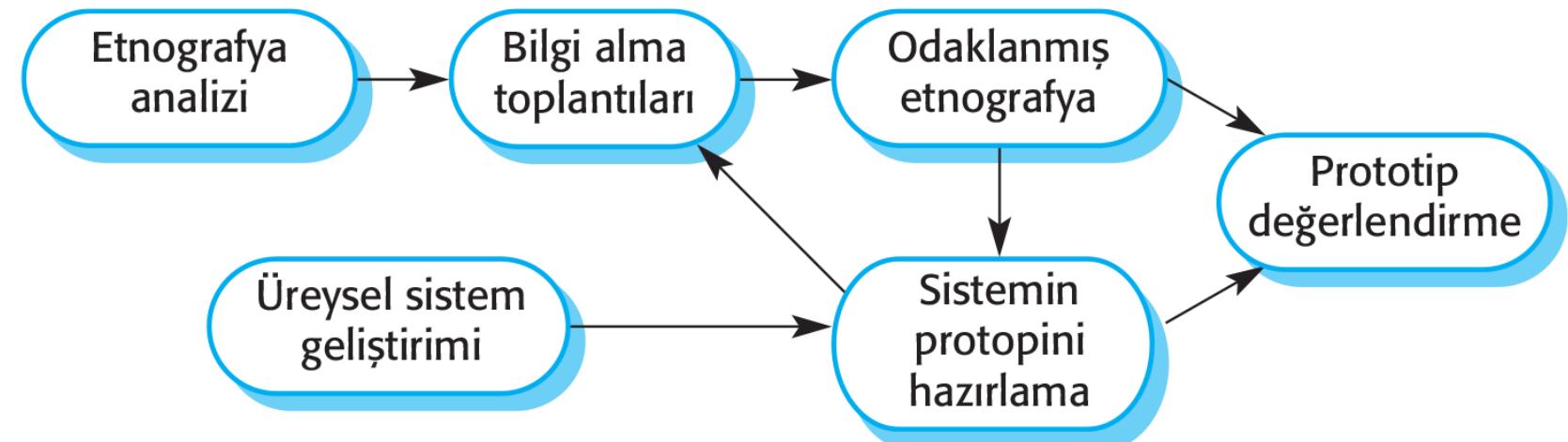
Şekil 4.7
Gereksinimlerin açığa çıkarılması ve analizi süreci



Bakış açıları

Bir bakış açısı ortak bir şeyleri olan bir grup paydaştan bir dizi gereksinimi toplama ve düzenlemenin bir yoludur. Bu nedenle, her bakış açısı bir dizi sistem gereksinimi içerir. Bakış açıları son kullanıcılardan, yöneticilerden ya da diğerlerinden gelebilir. Gereksinimleri konusunda bilgi sağlayabilecek kişilerin belirlenmesinde ve gereksinimlerin analiz için yapılandırılmasında yararlı olurlar.

Şekil 4.8
Gereksinimlerin analizi
için etnografa ve
prototip hazırlama



Sınıfta resim paylaşmak

Jack Ullapool'da (kuzey Scotland'da bir köy) bir ilkokul öğretmenidir. Bölgedeki balıkçılığın tarihine, gelişimine ve ekonomik etkisine bakarak, bir sınıf projesinin balıkçılık endüstrisi üzerine yoğunlaşmasına karar vermiştir. Bu projenin bir parçası olarak çocuklara akrabalarından anılar toplamaları ve paylaşmaları, gazetelerin arşivlerini kullanmaları, balıkçılık ve bölgedeki balıkçı toplulukları ile ilgili eski fotoğraflar toplamaları söylenir. Çocuklar birlikte balıkçılık hikâyelerini toplamak için eOgren, viki ve gazete arşivlerine ve fotoğraflarına ulaşmak için SCRAN (tarih kaynakları sitesi) kullanır. Ancak, Jack çocukların birbirlerinin resimlerini çekmesini ve yorumlamasını ve ailelerinden bulabilecekleri eski fotoğrafları tarayıp yüklemelerini istediği için bir fotoğraf paylaşım sitesine gereksinim duyar.

Jack üyesi olduğu ilkokul öğretmenleri grubuna herhangi biri uygun bir sistem önerebilecek mi diye bir elektronik posta yollar. İki öğretmen cevap verir ve her ikisi de öğretmenlerin içeriğini denetlemesine ve yönetmesine izin veren KidsTakePics fotoğraf paylaşım sitesini kullanmasını önerir. KidsTakePics eOgren kimlik belirleme servisi ile bütünleştirilmediği için Jack bir öğretmen ve sınıf hesabı yaratır. Sınıftaki çocuklar oturumlarını açtıklarında taşınabilir cihazlarından ve sınıf bilgisayarlarından fotoğrafları hemen yükleyebilmeleri için, eOgren kurulum servisini kullanarak, KidsTakePics'i çocuklar tarafından görülen servislere ekler.

Şekil 4.9

eOgren
sistemi için
kullanıcı
hikayesi



GEREKSİNİMLERİN SPESİFİKASYONU

Gereksinimlerin spesifikasyonu kullanıcı ve sistem gereksinimlerini bir gereksinim dokümanı içinde yazma sürecidir. İdeal olarak, kullanıcı ve sistem gereksinimleri açık, kesin, kolay anlaşılır, tam ve tutarlı olmalıdır. Pratikte, bunu elde etmek neredeyse imkânsızdır. Paydaşlar gereksinimleri farklı şekillerde yorumlar ve çoğu zaman gereksinimlerde içsel çelişkiler ve tutarsızlıklar vardır.

Kullanıcı gereksinimleri gereksinim dokümanında neredeyse her zaman uygun diyagramlar ve tablolar ile desteklenmiş doğal dilde yazılır. Sistem gereksinimleri de doğal dilde yazılabılır, ancak formlar, grafiksel ya da matematiksel sistem modelleri üstüne kurulu diğer gösterimler de kullanılabilir. Şekil 4.11 sistem gereksinimlerini yazmak için olası gösterimleri özetler.



Gösterim	Tanım
Doğal dil cümleleri	Gereksinimler numaralanmış cümleler kullanılarak doğal dilde yazılır. Her cümle bir gereksinimi açıklamalıdır.
Yapısal doğal dil	Gereksinimler doğal dilde standart bir form ya da şablon üzerine yazılır. Her alan gereksinimin bir yönü ile ilgili bilgi sağlar.
Grafik gösterim	Metin açıklamaları ile desteklenmiş grafik modeller sistemin fonksiyonel gereksinimlerini tanımlamak için kullanılır. UML (Unified modeling language) kullanma durumu ve sıra diyagramları sıkılıkla kullanılır.
Matematiksel spesifikasyonlar	Bu gösterimler sonlu makineler ya da kümeler gibi matematiksel kavramlara dayanır. Bu kesin spesifikasyonlar bir gereksinim dokümanında belirsizliği azaltsa da, pek çok müşteri resmi bir spesifikasyonu anlamaz. Bunun istediklerini ifade ettiğini kontrol edemezler ve onu bir sistem sözleşmesi olarak kabul etmeye isteksizdirler. (Bu yaklaşımı sistem güvenilebilirliğini içeren Bölüm 10'da tartışıyorum.)

Şekil 4.11 Sistem gereksinimlerini yazmak için gösterim



Şekil 4.12

İnsülin
pompası
yazılım
sistemi için
örnek
gereksinimler

3.2 Sistem kan şekerini ölçmeli ve eğer gerekliyse her 10 dakikada insülin sağlamalıdır. (*Kan şekerinde değişimler oldukça yavaştır, bu nedenle daha sık ölçüm gereksizdir; daha seyrek ölçüm gereksiz yüksek şeker düzeylerine neden olabilir.*)

3.6 Sistem her dakika denenecek koşullar ve Tablo 1'de tanımlanan karşılık gelen eylemler ile bir kendi
dine deneme yordamı çalışmalıdır. (*Kendini deneme yordamı donanım ve yazılım problemlerini bulabilir ve kullanıcıyı normal işlevin olanaksızlığı ile ilgili uyarabilir.*)

**Gereksinim spesifikasyonu için doğal dil kullanmanın problemleri**

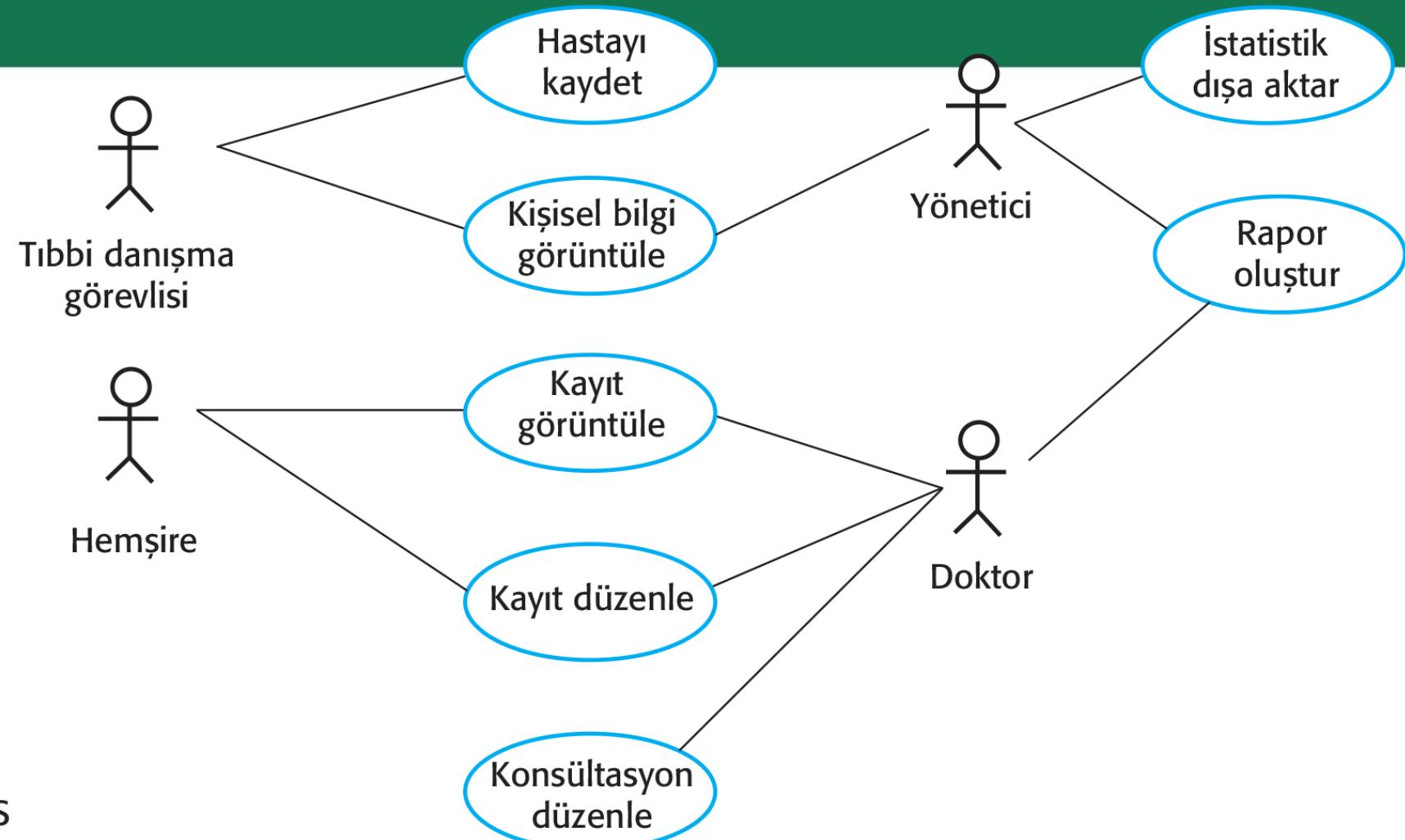
Spesifikasyon için yararlı olan doğal dil esnekliği sıkça problemlere neden olur. Açık olmayan gereksinimler yazmak olanaklıdır ve okuyucular (tasarımcılar) kullanıcıdan farklı bir geçmişe sahip oldukları için gereksinimleri yanlış yorumlayabilir. Birkaç gereksinimi tek cümlede birleştirmek kolaydır ve doğal dil gereksinimleri yapılandırmak zor olabilir.

İnsülin Pompası/Denetleme Yazılımı/SRS/3.3.2

Fonksiyon	İnsülin dozunu hesapla: Güvenli şeker düzeyi.
Tanım	O an ölçülen şeker düzeyi 3 ile 7 birim arasındaki güvenli dilimdeyken verilmesi gereken insülin dozunu hesaplar.
Girdiler	O anki şeker okuması (r_2), önceki iki okuma (r_0 ve r_1).
Kaynak	Sensörden gelen o anki şeker okuması. Bellekten gelen diğer okumalar.
Çıktılar	CompDose-Verilmesi gereken insülin dozu.
Hedef	Ana denetleme döngüsü.
Eylem	Şeker düzeyi kararlıysa ya da düşüyorsa ya da düzey artıyor ama yükselme hızı düşüyorsa CompDose sıfırdır. Eğer düzey artıyorsa ve artma hızı artıyorsa CompDose o anki ve bir önceki şeker düzeylerinin farkının 4'e bölünmesi ve sonucun yuvarlanması ile hesaplanır. Eğer sonuç sıfıra yuvarlanıyorsa CompDose verilebilecek en az doza ayarlanır (Şekil 4.14'e bakınız).
Gerektirir	Şeker düzeyi değişim hızının hesaplanabilmesi için önceki iki okuma.
Ön koşul	İnsülin deposu en az tek dozda en çok verilebilecek doz kadar insülin içerir.
Son koşul	r_0 r_1 ile değiştirilir sonra r_1 r_2 ile değiştirilir.
Yan etkileri	Yok

Şekil 4.13 Bir insülin pompası için yapısal spesifikasyon





Şekil 4.15 ZihinSaS sistemi için kullanım durumları

GEREKSİNİMLERİN DOĞRULANMASI

Gereksinimlerin doğrulanması müşterinin gerçekten istediği sistemi tanımlayan gereksinimlerin kontrol edilmesi sürecidir. Gereksinim problemlerinin bulunması ile ilgilendiği için açığa çıkarma ve analiz ile örtüşür. Gereksinimlerin doğrulanması süreci sırasında gereksinim dokümanındaki gereksinimler üzerinde farklı tip kontroller uygulanmalıdır. Bu kontroller aşağıdakileri içerir:

**1. Geçerlik
Kontrolleri**

**2. Tutarlılık
Kontrolleri**

**3. Tamlik
Kontrolleri**

**4. Gerçekçilik
Kontrolleri**

**5. Doğrulanabilirlik
Kontrolleri**





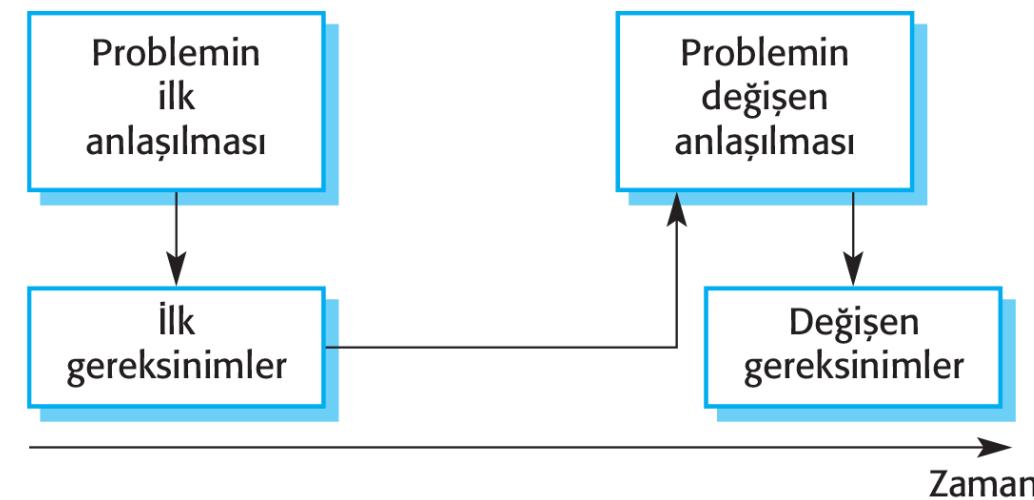
Gereksinim incelemeleri

Gereksinim incelemesi sistem müşterilerinden ve geliştiricilerinden bir grup insanın gereksinim dokümanını ayrıntılı olarak okuması ve hataları, anormallikleri ve tutarsızlıklarını araması sürecidir. Bunlar saptandığında ve kayıt edildiğinde nasıl çözüleceklerini tartışmak müşteri ve geliştiriciye kalır.

GEREKSİNİMLERİN DEĞİŞİMİ

Büyük yazılım sistemleri için gereksinimler her zaman değişir. Bu sistemlerin genellikle “şeytani” problemleri (bütünyle tanımlanamayan problemleri) adreslemek üzere geliştiriliyor olması sık değişikliklerin bir nedenidir.

Problemler tam olarak tanımlanamadıkları için, yazılım gereksinimlerinin eksik olması kesin gibidir. Yazılım geliştirimi süreci sırasında paydaşların problemi anlayışları sürekli değişir (Şekil 4.18). Bu durumda sistem gereksinimleri bu problem anlayışındaki değişikliği yansıtmak için değiştirmek zorundadır.

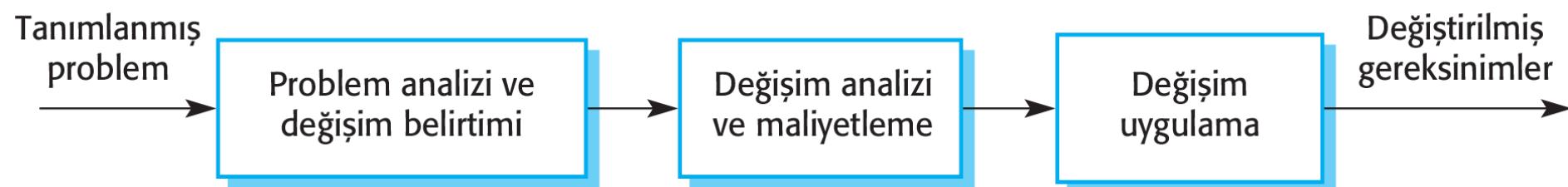


Şekil 4.18
Gereksinimlerin evrimi



Kalıcı ve uçucu gereksinimler

Bazı gereksinimler değişime diğerlerinden daha duyarlıdır. Kalıcı gereksinimler organizasyonun ana, yavaş değişen eylemleri ile ilişkilidir. Uçucu gereksinimler temel iş eylemleri ile ilişkilidir. Uçucu gereksinimler büyük olasılık ile değişir. Bunlar genellikle işin kendisinden çok organizasyonun o işi nasıl yaptığı yansıtan destekleyici eylemler ile ilişkilidir.



Şekil 4.19
Gereksinim değişimi
yönetimi

3.Hafta

İsterlerin Belirlenmesi



İÇERİK

Bu bölümde,

- İsterlerin belirlenmesi aşamasının genel hatları ve diğer yazılım geliştirme faaliyetleri ile olan ilişkisi,
- İsterlerin belirlenmesi aşamasında kullanılan temel kavramlar ve terimler,
- İsterlerin belirlenmesi aşamasındaki faaliyetler,
- İsterlerin belirlenmesi ile ilgili faaliyetlerin kontrolü ve yönetimi konuları anlatılacaktır.



Yazılım İsterleri Çözümlemesi

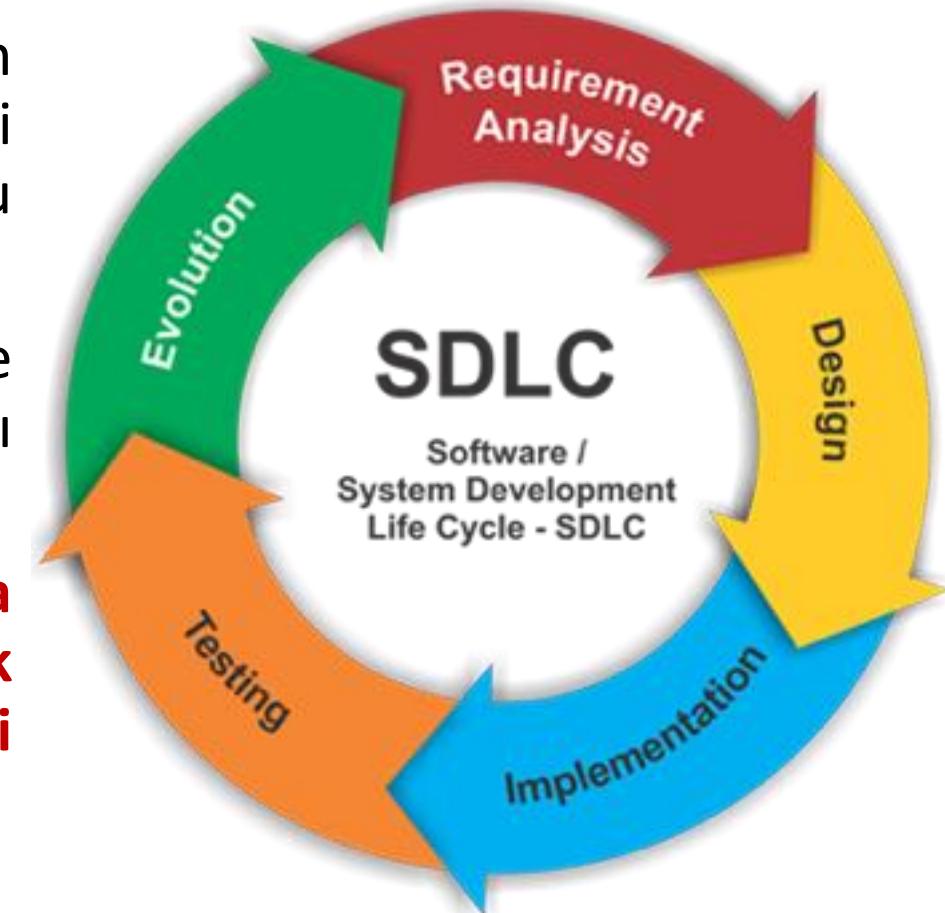
Yazılım gereksinimi karşılanması gereken bir nesnedir. Planlamacılar gereksinimlerin çogunu fonksiyonel olarak karşılayarak tasarım ve implementasyon ile ilgili ayrıntıları geliştircilere bırakırlar. Maliyet, performans ve güvenilirlikle ilgili amaçlar kullanıcı arayüzleri doğrultusunda ayrıntılı olarak planlamacılar tarafından gerçekleştirilmelidir. Bazen planlamacılar amaçlarını gerçekçi olmak yerine kurala bağlı olarak tanımlarlar.

Kaner et. al., «Testing Computer Software», 1996



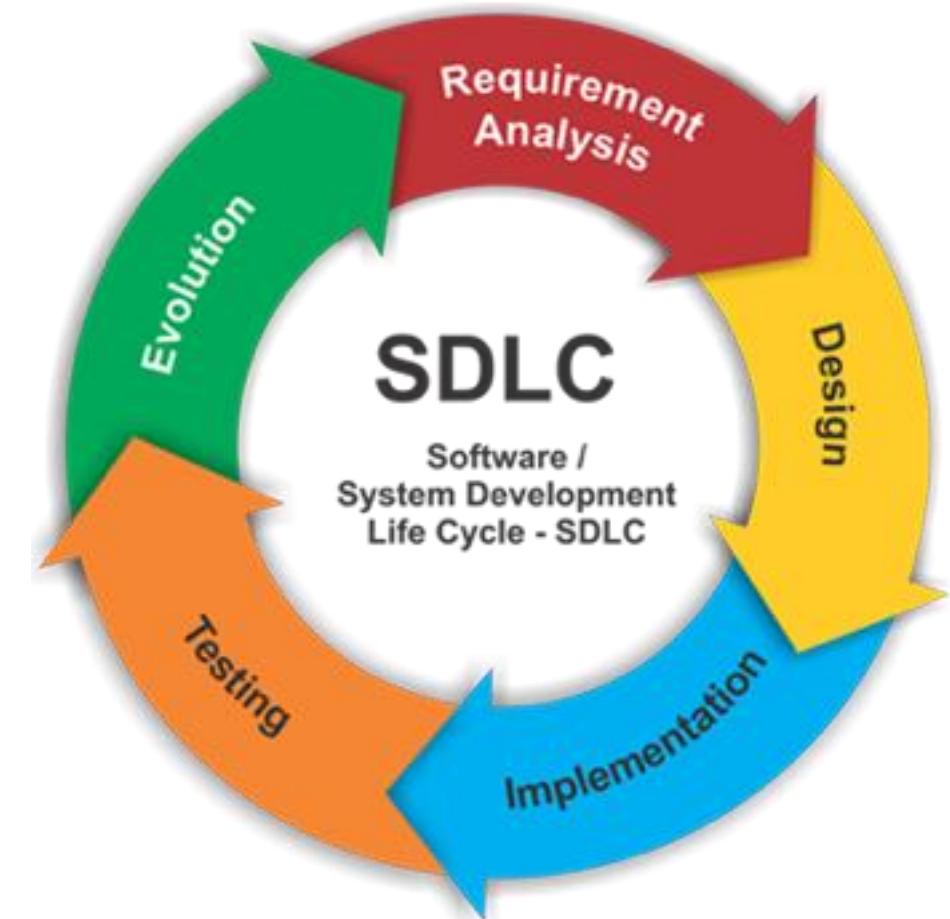
Yazılım İsterleri Çözümlemesi

- Yazılım geliştirme sürecinin başarısı için gereksinim analizi (requirements analysis) ya da yazılım isterleri çözümlemenin (software requirements analysis) doğru yapılması son derece önemlidir.
- Bir yazılım ne kadar iyi tasarılanırsa ya da geliştirilirse geliştirilsin müşteri (ya da son kullanıcı) ihtiyaçlarını karşılamıyorsa başarılı sayılamaz.
- **Yazılım isterlerinin çözümlenmesi aşamasında müşterinin yazılımdan ne beklediği ayrıntılı olarak belirlenir, gereksinimler açığa çıkarılır, yazılım isterleri modellenir ve tanımlanarak tasarım aşamasına geçilir.**



Yazılım İsterleri Çözümlemesi

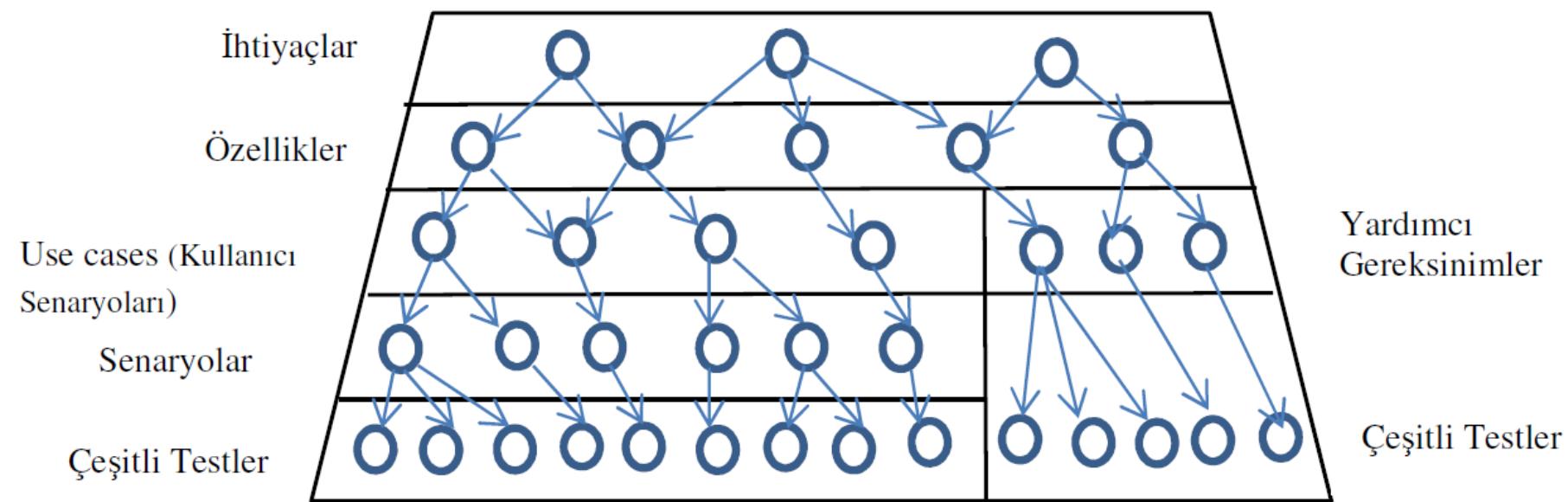
- Yazılım isterlerinin çözümlenmesi aşaması müşteri ve geliştirme grubu arasında ciddi bir işbirliği gerektirir.
- Bu konudaki standartlar: IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.
- IEEE/EIA 12207 Standard for Information Technology-software life cycle processes-Life cycle data.



Yazılım İsterleri Çözümlemesi

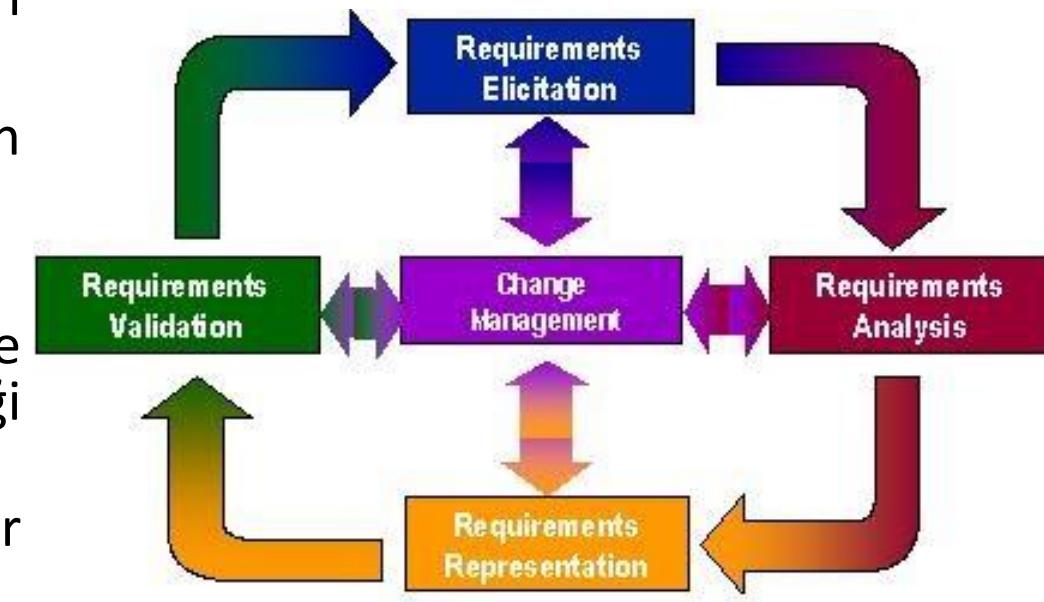
- Bir yazılım ürünü için çözümünün baslangıcı olan gereksinimlerin yönetimi de önemlidir.
- Müsteriler, geliştiriciler, proje sahibi ve diğer istirakçilerin sürekli etkileşimde olmaları tasarım hatalarını önlemede önemlidir. Ürün ile ilgili olarak geliştirilen bir yol haritası, teslimat sonrasında müşterinin ürünü olan güvenini arttıracaktır.

Altan, Z., The Importance Of Requirements Engineering On Software Products Development,



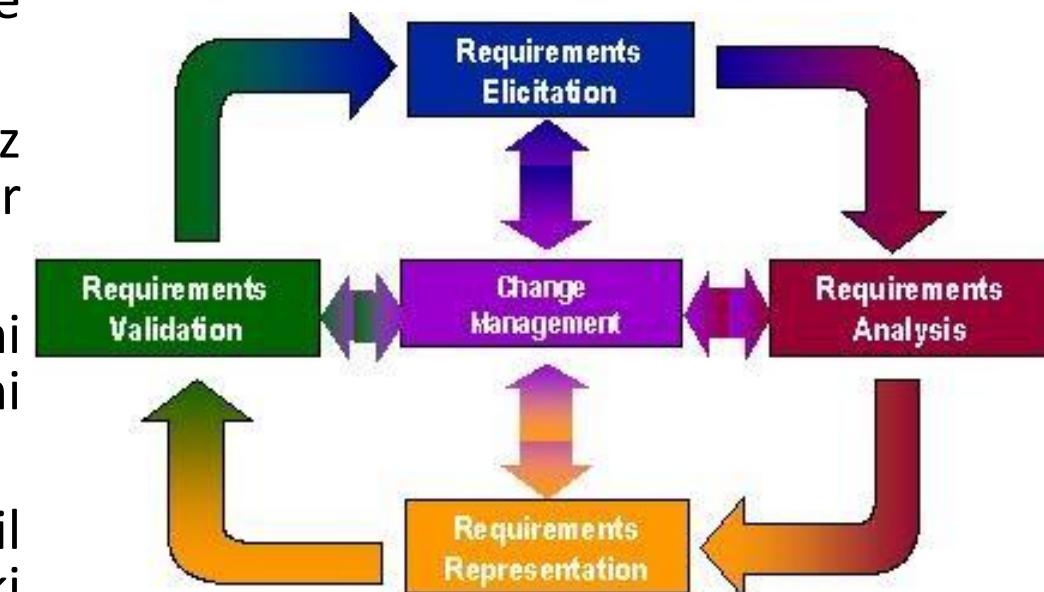
Yazılım İsterleri Çözümlemesi

- **Gereksinim**, sistemin sahip olması gereken bir özellik veya istemci tarafından kabul edilmek için karşılaması gereken bir kısıtlamadır.
- Gereksinim mühendisliği, yapım aşamasında olan sistemin gereksinimlerini tanımlamayı amaçlamaktadır.
- Gereksinim mühendisliği iki ana faaliyet içerir;
 - Müşterinin anladığı sistemin spesifikasyonuna ve geliştiricilerin açık bir şekilde yorumlayabileceği gereksinim özelliklerinin ortaya çıkarılması.
 - Geliştiricilerin açık bir şekilde yorumlayabileceği bir analiz modelinin oluşturulması
- Gereksinimlerin ortaya çıkarılması ikisinden daha zor olanıdır çünkü farklı geçmişlere sahip birkaç katılımcı grubunun işbirliğini gerektirir.

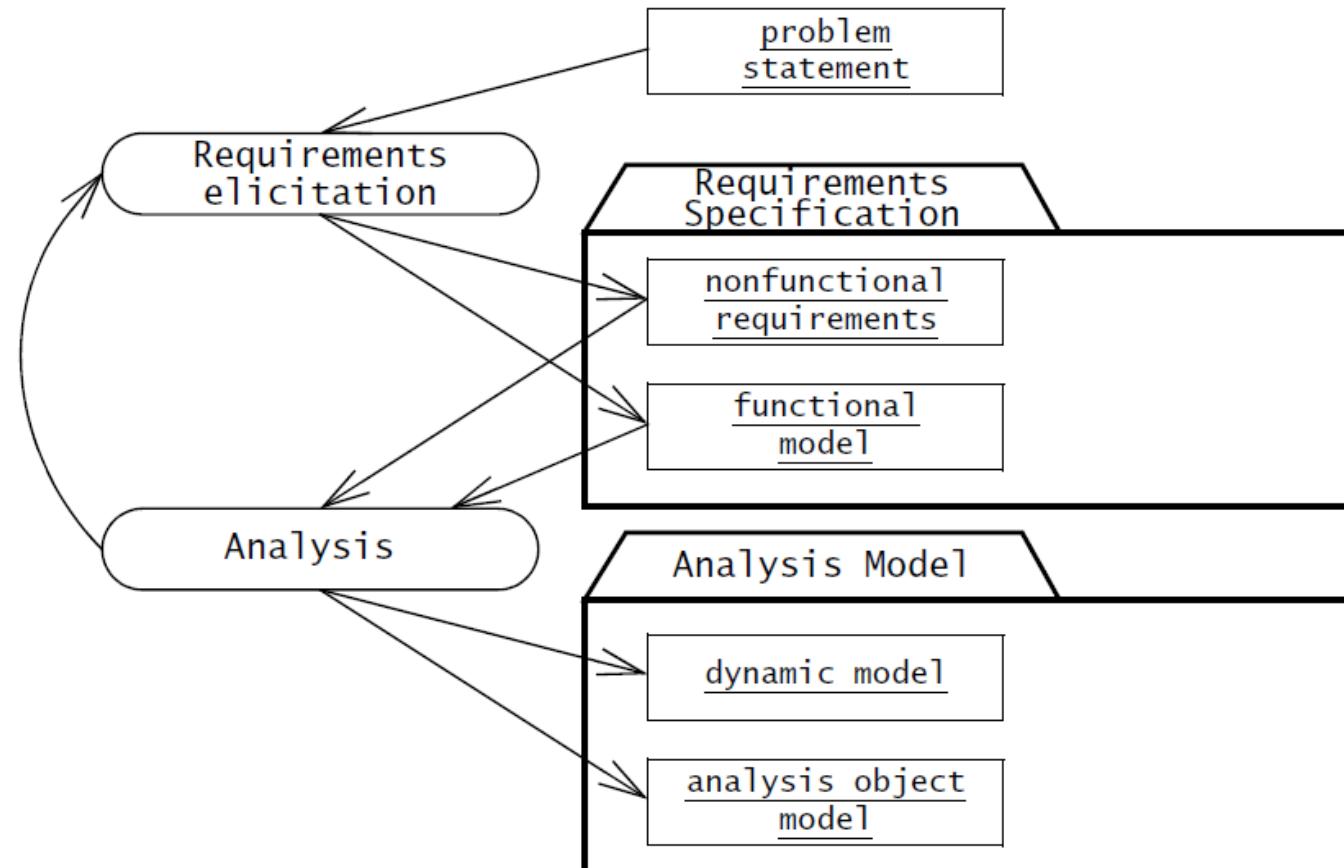


Yazılım İsterleri Çözümlemesi

- Hem gereksinim özelliklerini hem de analiz modeli aynı bilgiyi temsil eder. Yalnızca kullandıkları dilde ve gösterimde farklılık gösterirler;
- Gereksinimler özelliği doğal dilde yazılırken, analiz modeli genellikle biçimsel veya yarı biçimsel bir gösterimle ifade edilir.
- Gereksinim özellikleri, müşteri ve kullanıcılarla iletişimini destekler. Analiz modeli, geliştiriciler arasındaki iletişimini destekler.
- Her ikisi de sistemin dış yönlerini doğru bir şekilde temsil etmeye çalışmaları anlamında sistemin modelidir. Her iki modelin de sistemin aynı yönlerini temsil ettiği göz önüne alındığında, gereksinimlerin ortaya çıkarılması ve analizi eşzamanlı ve yinelemeli olarak gerçekleşir.

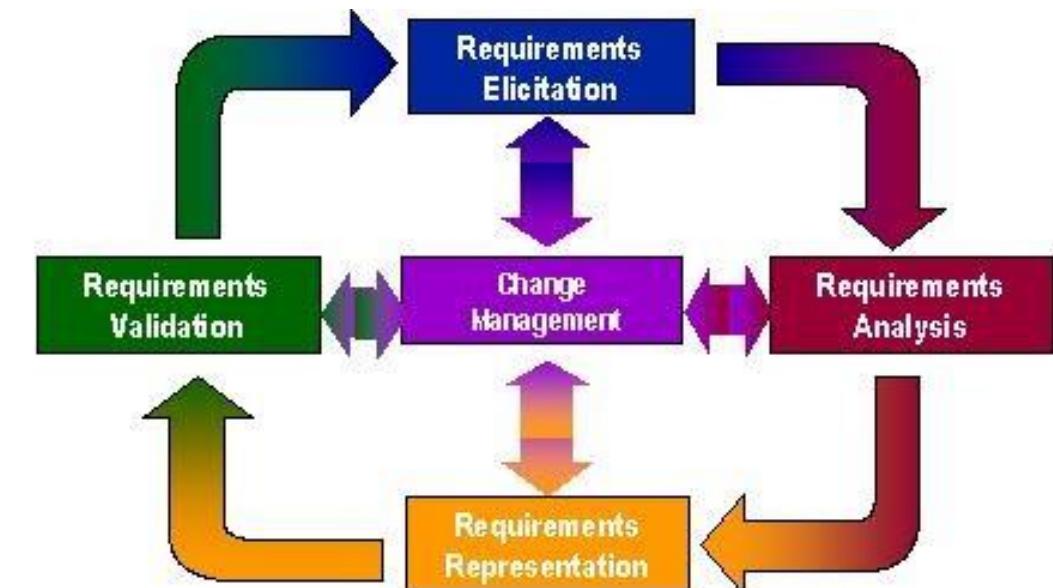


Yazılım İsterleri Çözümlemesi



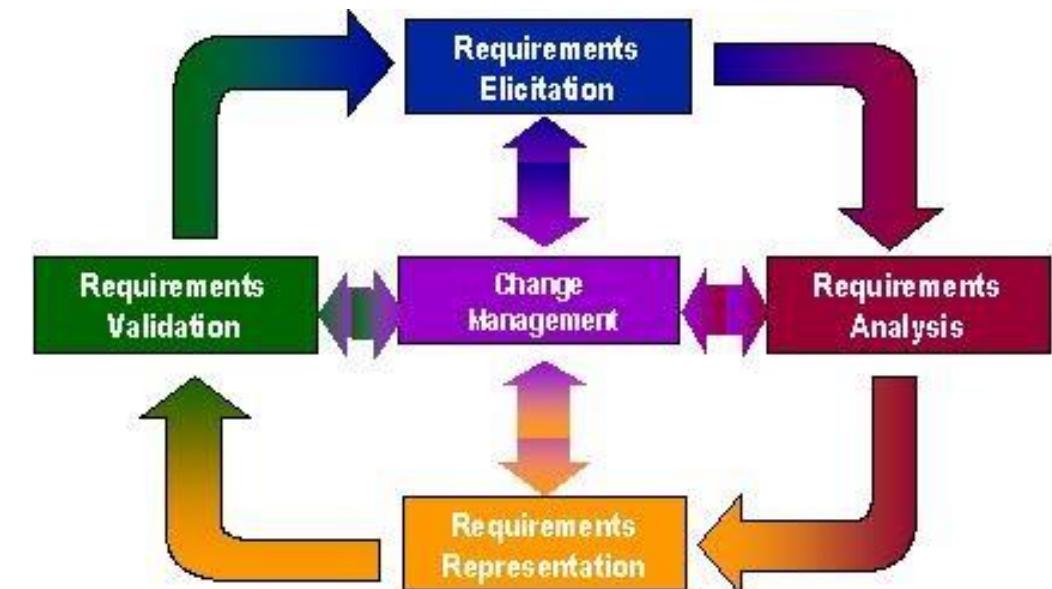
İSTERLERİN BELİRLENMESİ (REQUIREMENTS ELICITATION)

- Müşteri ve kullanıcılar sistemin uygulama alanı ve sahip olması gereken fonksiyonları iyi bilirken, sistem yazılımının geliştirilmesi konusunda çok az bilgi ve deneyime sahiptirler.
- Diğer taraftan, sistem geliştiricileri de sistem yazılımının geliştirilmesi konusunda çok deneyimli ve bilgili iken günlük uygulama alanı konusunda az bilgi ve deneyime sahiptirler.
- Senaryo ve Kullanım Durumları (Use Case)** kullanıcılar ve sistem geliştiricileri arasındaki iletişime ve bilgi alışverişine olanak veren araçlardır. Senaryo, sistemin kullanımına örnek olarak sistem ve kullanıcı arasındaki bir dizi etkileşimi tanımlar.



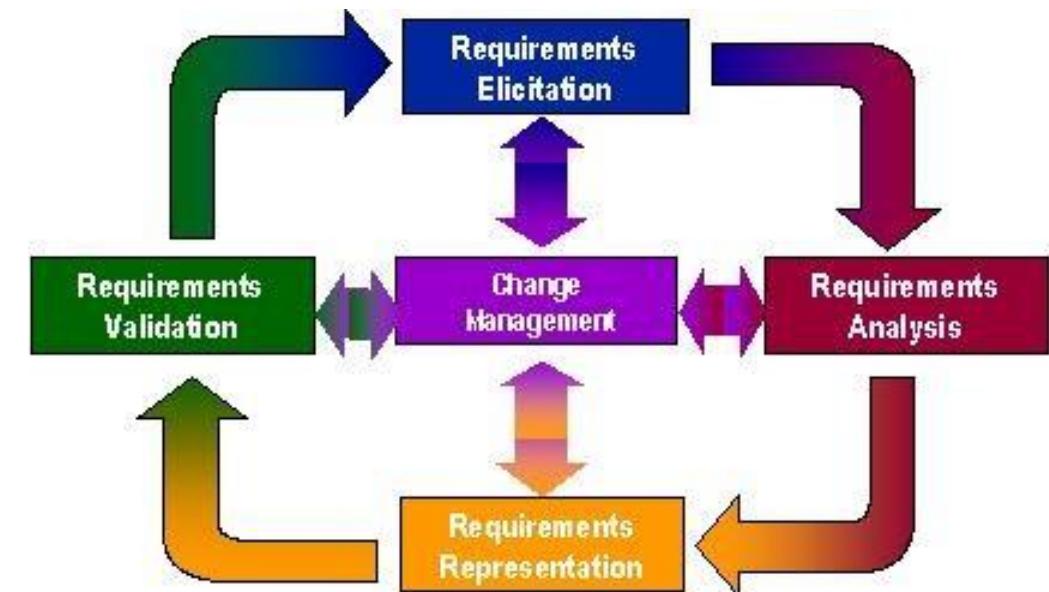
İSTERLERİN BELİRLENMESİ (REQUIREMENTS ELICITATION)

- Kullanım Durumu ise bir Senaryo sınıfını tanımlayan bir **soyutlama (abstraction)** dır. Hem senaryo hem de Kullanım Durumları kullanıcının rahatlıkla anlayabileceği bir dilde yazılırlar.
- Bu bölümde, isterlerin senaryolara dayanarak belirlenmesi üzerinde duracağız. Sistem geliştiricileri, isterleri gözlemlerine ve kullanıcılarla olan mülakatlarına dayanarak belirlerler.
- Geliştiriciler, önce kullanıcının iş proseslerini **ış-ış senaryoları** şeklinde betimlerler, sonra da geliştirilecek olan sistemin işlevsellliğini **hayali (visionary) senaryolar** oluşturarak tanımlarlar.



İSTERLERİN BELİRLENMESİ (REQUIREMENTS ELICITATION)

- Müşteri ve kullanıcılar, sistem tanımının geçerliliğini senaryoları gözden geçirerek ve geliştiriciler tarafından sunulan küçük prototipleri test ederek belirlerler.
- Sistemin tanımı belirgin ve istikrarlı bir duruma gelince müşteri ve sistem geliştiricileri Kullanım Durumu formunda bir sistem spesifikasyonu üzerinde anlaşmaya varırlar.



Kullanılabilirlik Örnekleri

- Sistemin kullanılabılırlığını arttırmak için, sistem tanımı oluşturularken hataların ve belirsizliklerin mümkün olduğunda önlenmesi çok önemlidir.
- Örneğin, ABD'de nokta ondalık ayıracı, virgül ise binlik ayıracı ifade eder. Almanya'da ise bunun tam tersi bir standart kabul edilmiştir. Almanya'daki bir kullanıcının her iki farklı standarttan da haberdar olduğunu düşünelim. Bu kullanıcı online bir katalogtaki fiyat listesini incelerken hangi standardın kullanılmış olduğunu bilmediğinden bir belirsizlikle karşılaşacaktır. Bu tip belirsizlikleri en aza indirmek için müşteri, kullanıcı ve sistem geliştiricileri arasındaki iletişimin çok etkili bir şekilde sağlanması gereklidir.



Kullanılabilirlik Örnekleri

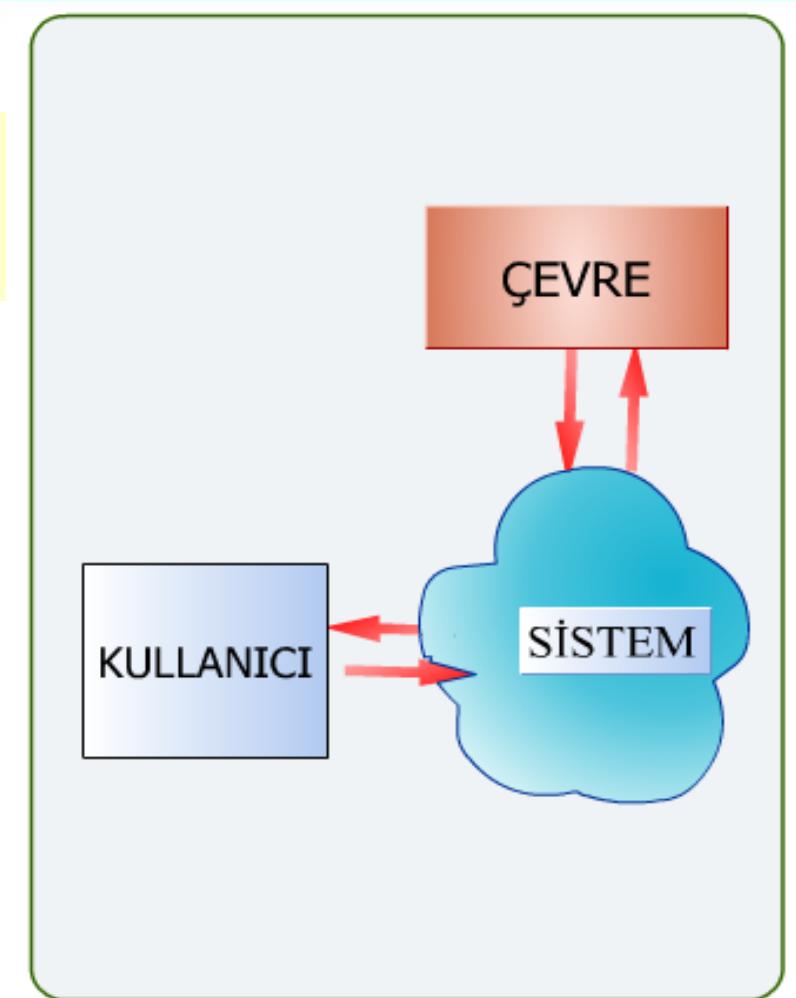
- İsterlerin belirlenmesi aşaması belirsizlikleri en aza indirmek için müşteri, kullanıcı ve sistem geliştiricileri arasındaki iletişim çok etkili bir şekilde sağlanmasını amaçlamaktadır.
- Bu aşamada oluşan hataları ve belirsizlikleri sonradan düzeltmeye çalışmak hem çok zor hem de çok pahalıdır.
- Bu hatalar, sistemin gereklili olan bir **işlevselligi (functionality)** sağlamamasına, gereksiz bir işlevselligi sağlamasına, ya da bir işlevselligi eksik ve hatalı bir şekilde sağlamasına yol açar.
- Bu nedenle isterlerin belirlenmesi aşaması sistem geliştiricisi, müşteri ve kullanıcılar tarafından çok dikkatli bir şekilde yürütülmelidir.



İsterlerin Belirlenmesi Aşamasının Genel Hatları

Müşteri ve sistem geliştiricileri sistemin spesifikasyonunu belirlerler ve bu spesifikasyon müşteri ve sistem geliştiricileri arasında bir kontrat niteliğini taşır.

- Sistem spesifikasyonu, isterlerin analizi aşamasında yapılandırılır ve biçimlendirilir.
- Bu aşamanın sonucunda sistemin analiz modeli oluşturulur. Hem sistem spesifikasyonu hem de analiz modeli aynı bilgiyi betimlemektedir, fakat betimleme şekilleri farklıdır.

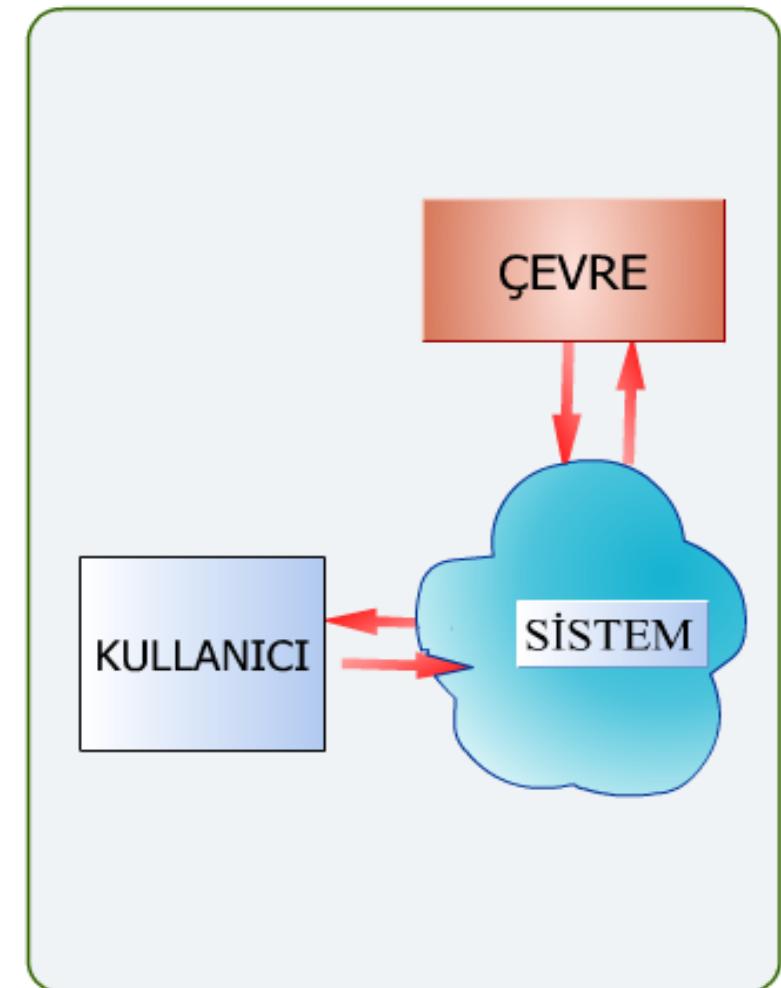


Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

Müşteri ve sistem geliştiricileri sistemin spesifikasyonunu belirlerler ve bu spesifikasyon müşteri ve sistem geliştiricileri arasında bir kontrat niteliğini taşır.

- İsterlerin belirlenmesi aşamasında sistem, kullanıcının bakış açısı ile ele alınır.
- Örneğin sistemin işlevselligi, kullanıcı ve sistem arasındaki etkileşim, sistemin saptayıp üstesinden gelebileceği hatalar ve sistemin çalışacağı çevre koşulları isterlerin bir kismıdır.
- Diğer taraftan sistemin yapısı, gerçekleştirilmesi için seçilen teknoloji, sistemin dizaynı gibi **doğrudan kullanıcı tarafından görülemeyen unsurlar** ister değildir.

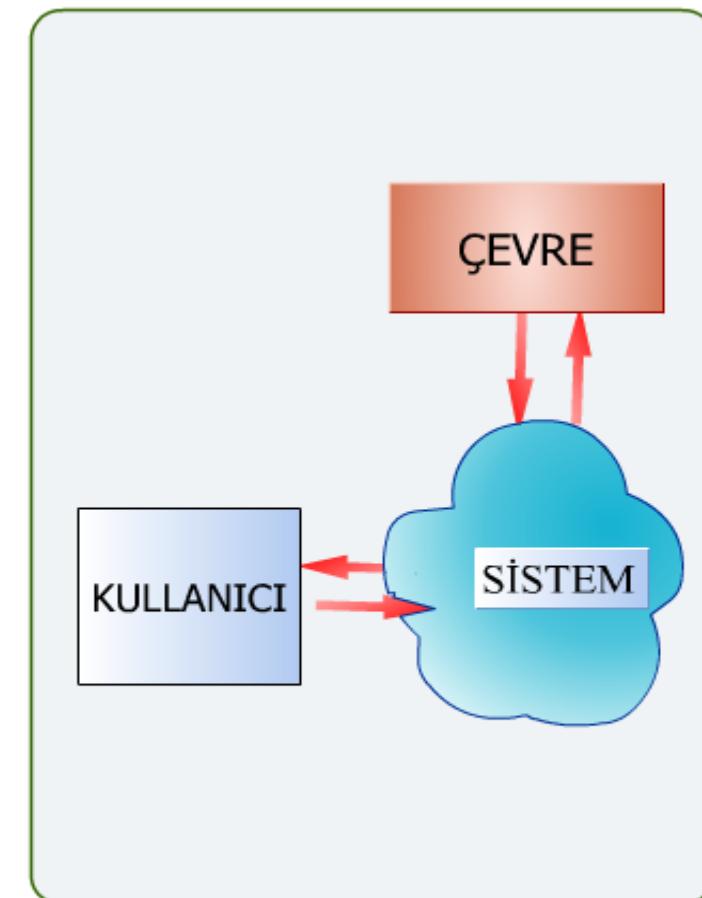


Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

İsterlerin belirlenmesi aşamasında izlenecek adımlar:

- Aktörlerin belirlenmesi
- Senaryoların belirlenmesi
- Kullanım Durumlarının belirlenmesi
- Kullanım Durumlarının iyileştirilmesi
- Kullanım Durumları arasındaki ilişkilerin belirlenmesi
- İşlevsel olmayan isterlerin belirlenmesi

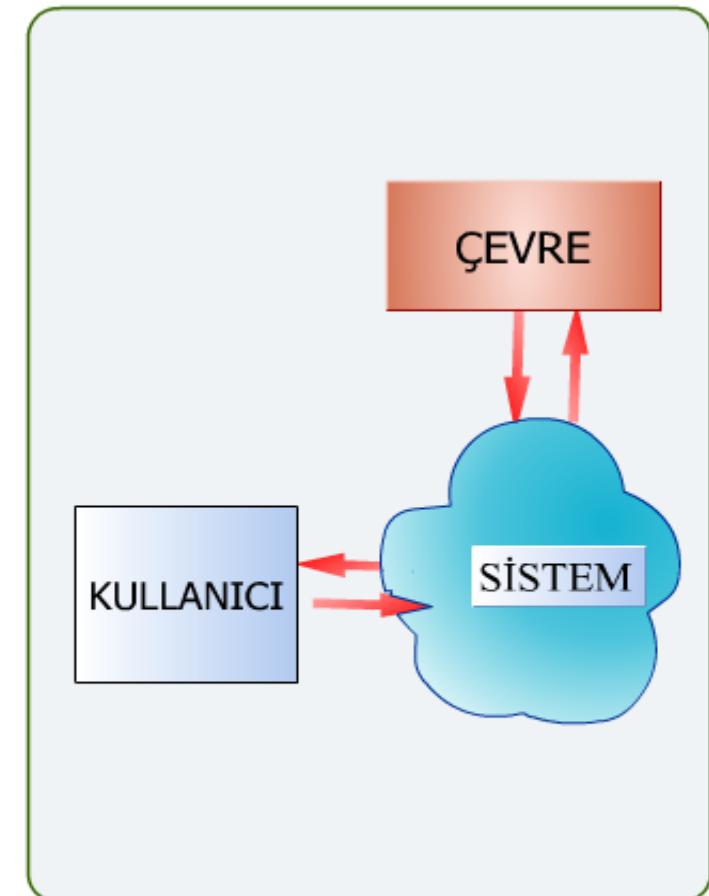


Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

İsterlerin belirlenmesi aşamasında izlenecek adımlar:

- **Aktörlerin belirlenmesi.** Bu etkinlik sırasında geliştiriciler, gelecekteki sistemin destekleyeceği farklı kullanıcı türlerini belirler.
- **Senaryoları tanımlama.** Bu etkinlik sırasında, geliştiriciler kullanıcıları gözlemler ve gelecekteki sistem tarafından sağlanan tipik işlevler için bir dizi ayrıntılı senaryo geliştirir. Senaryolar, kullanılmakta olan gelecekteki sistemin somut örnekleridir. Geliştiriciler bu senaryoları kullanıcıyla iletişim kurmak ve uygulama etki alanı hakkındaki anlayışlarını derinleştirmek için kullanır.
- **Kullanım durumlarının belirlenmesi.** Geliştiriciler ve kullanıcılar bir dizi senaryo üzerinde anlaştıktan sonra, geliştiriciler senaryolardan gelecekteki sistemi tamamen temsil eden bir dizi kullanım durumu çıkarırlar. Senaryolar tek bir durumu gösteren somut örnekler iken, kullanım durumları tüm olası durumları açıklayan soyutlamalardır. Kullanım durumlarını açıklarken, geliştiriciler sistemin kapsamını belirler.

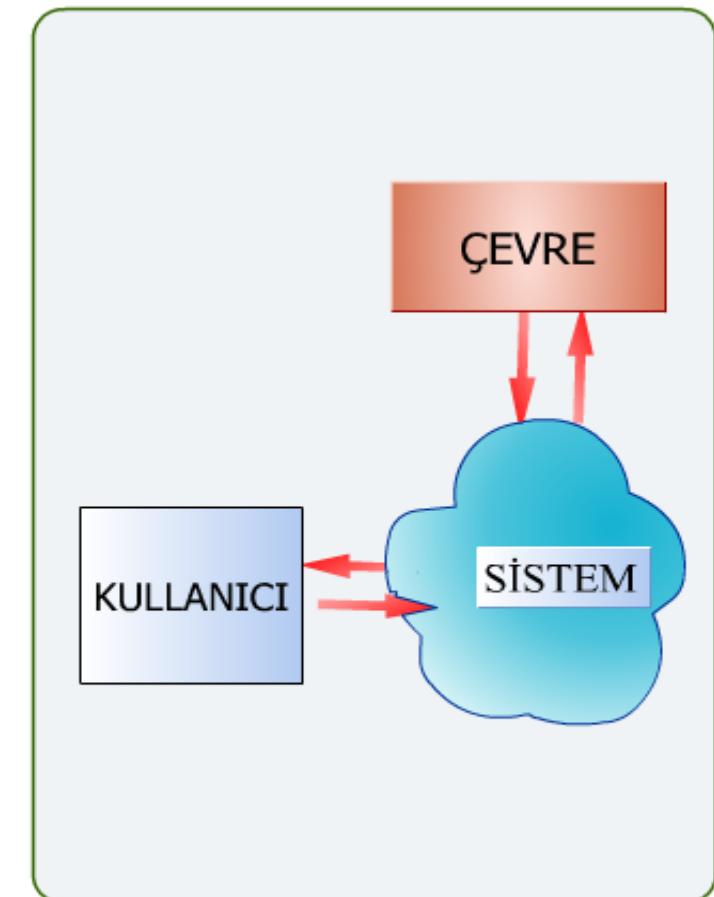


Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

İsterlerin belirlenmesi aşamasında izlenecek adımlar:

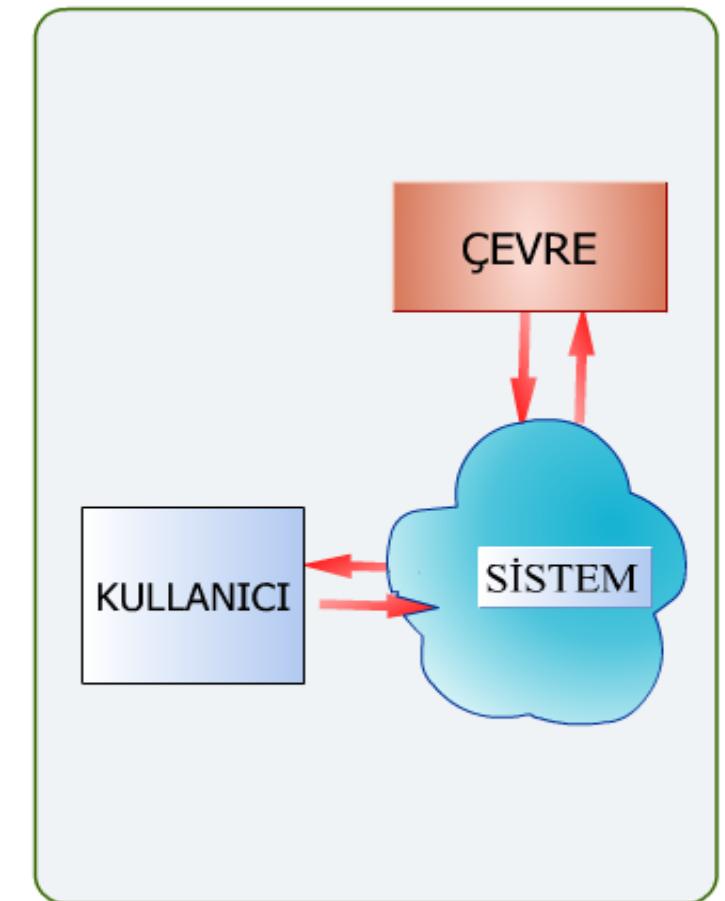
- Kullanım durumlarını iyileştirme.** Bu etkinlik sırasında geliştiriciler, her bir kullanım durumunu detaylandırarak ve hatalar ve istisnai koşullar durumunda sistemin davranışını açıklayarak gereksinim spesifikasyonunun eksiksiz olmasını sağlar.
- Kullanım durumları arasındaki ilişkileri belirlemek.** Bu etkinlik sırasında geliştiriciler, kullanım durumları arasındaki bağımlılıkları belirler. Ayrıca, ortak işlevselligi hesaba katarak kullanım durumu modelini konsolide ederler. Bu, gereksinim spesifikasyonunun tutarlı olmasını sağlar.
- İşlevsel olmayan gereksinimleri tanımlama.** Bu etkinlik sırasında geliştiriciler, kullanıcılar ve müşteriler, kullanıcı tarafından görülebilen, ancak işlevsellikle doğrudan ilgili olmayan hususlar üzerinde anlaşırlar. Bunlar, sistemin performansı, dokümantasyonu, tükettiği kaynaklar, güvenliği ve kalitesi üzerindeki kısıtlamaları içerir.



Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

- İsterlerin belirlenmesi aşamasında sistem geliştiricileri, müşteri tarafından sunulan sistemin uygulama alanı ile ilgili dokümanlar, kullanım kılavuzları ve teknik dokümanlar gibi birçok kaynaktan bilgi edinmek zorunda kalırlar.
- Bu kaynaklardan en önemlileri müşteri ve kullanıcıların kendileridir. Müşteri ve kullanıcılarla sistem geliştiricileri arasındaki bilgi alışverişini ve iletişimini sağlamaya yönelik üç yöntem üzerinde duracağız:
 - Joint Application Design (JAD)
 - Knowladge Analysis of Task (KAT)
 - Usability Testing

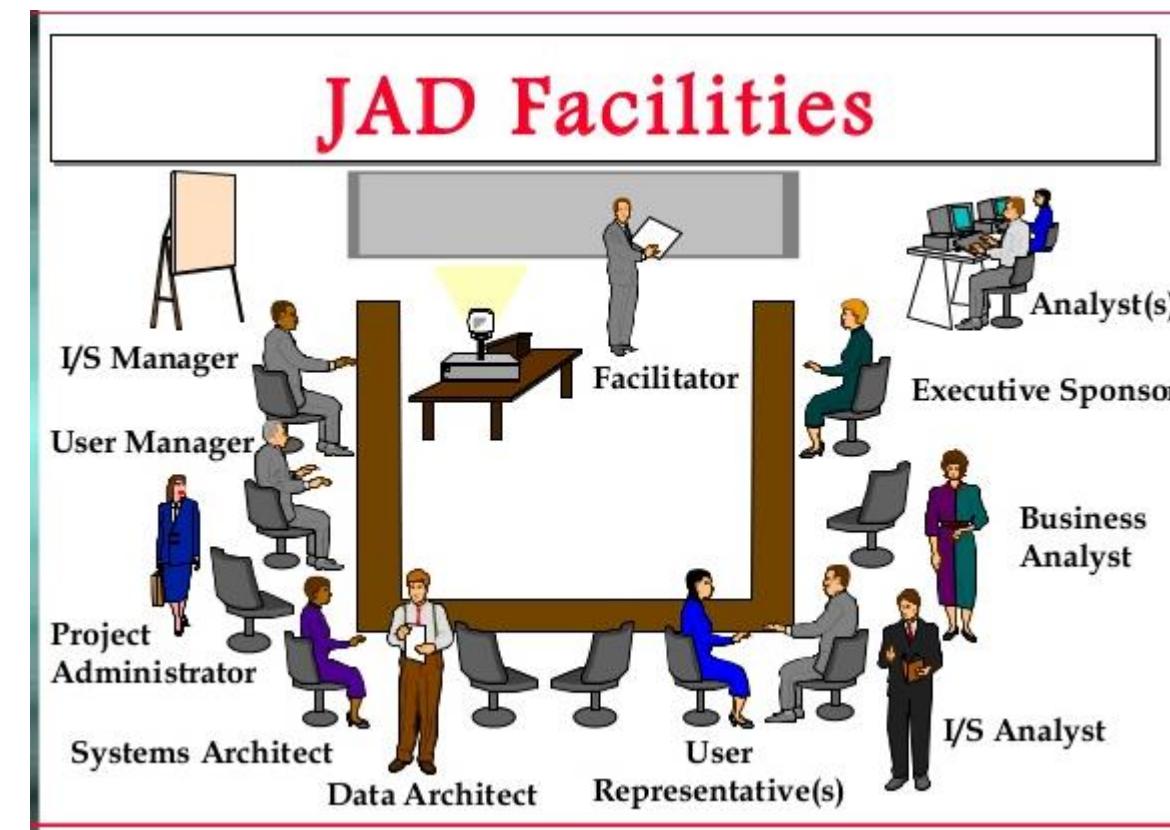


Sistemin çevre kullanıcılar ile olan ilişkileri, isterlerin belirlenmesinde kullanılan unsurlardır.

İsterlerin Belirlenmesi Aşamasının Genel Hatları

JAD (Joint Application Design)

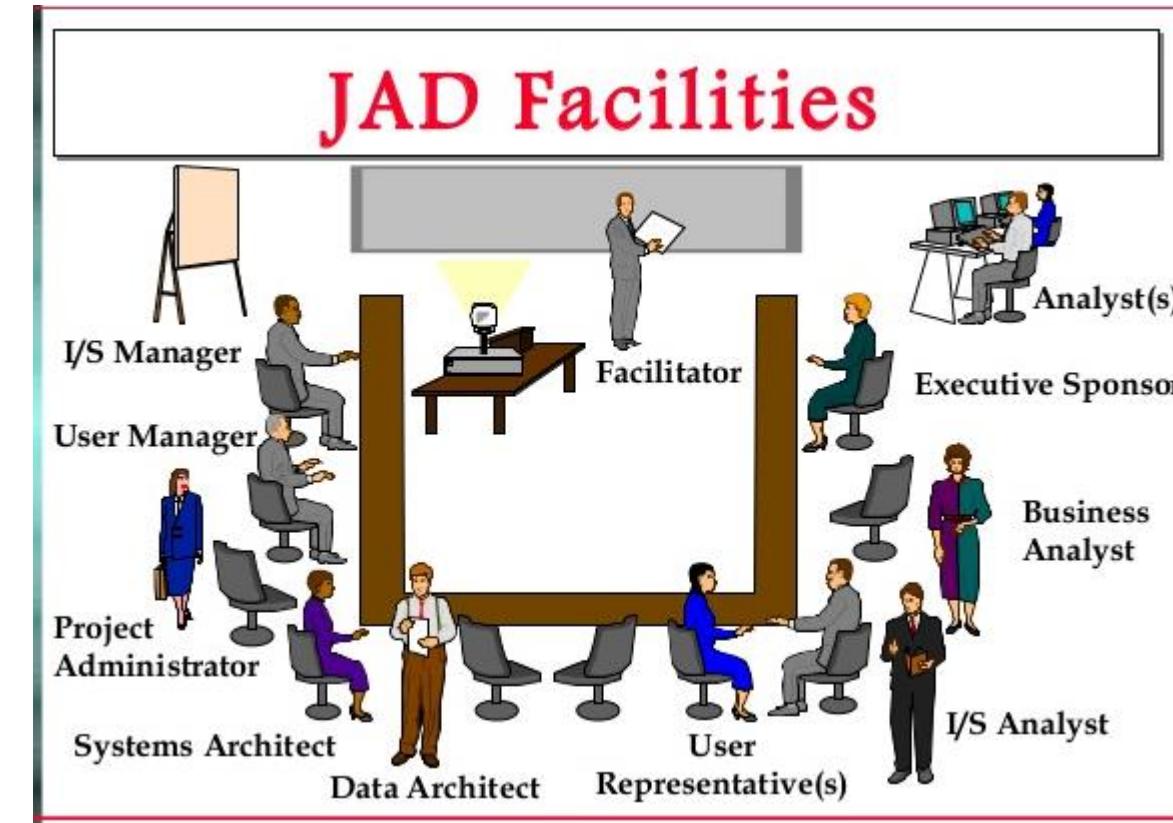
- JAD, yeni bilgi sistemlerinin geliştirilmesi sırasında gereksinimlerin toplanması için uygulanan ve iteratif geliştirme yaklaşımının önemli parçalarından biri olan hızlandırıcı çözüm tekniğidir.
- Amacı; fikir birliği ile kabul edilmiş sistem gereksinimlerini ortaya çıkarmak maksadıyla yapılandırılmış Workshop'lar ile geliştirme aşamasında yer alacak olan IT ve Business tarafını bir arada tutmak olan JAD, Tony Crawford ve Chuck Morris (IBM) tarafından 1977'de dağıtık sistemlere ait gereksinimleri elde etme metodu olarak tasarlanmıştır. 1980 yılında IBM Canada JAD yaklaşımını kabul etmiş ve işlemiştir.



İsterlerin Belirlenmesi Aşamasının Genel Hatları

JAD (Joint Application Design)

- JAD metodunun yıllar içerisinde prototipleme elementlerini, Case kavramını içinde barındırmak üzere evrimleşmeye devam etmesi üzerine bazı insanlar artık Jad'ın sadece bir yaklaşım değil bütün bir geliştirme süreci olduğunu düşünmeye başlamış ve "Joint Application Development" olarak adlandırmaya başlamışlardır. Fakat ne yazık ki JAD, tanımlama, analiz ve tasarım alanlarında biçimlendirilmiştir.

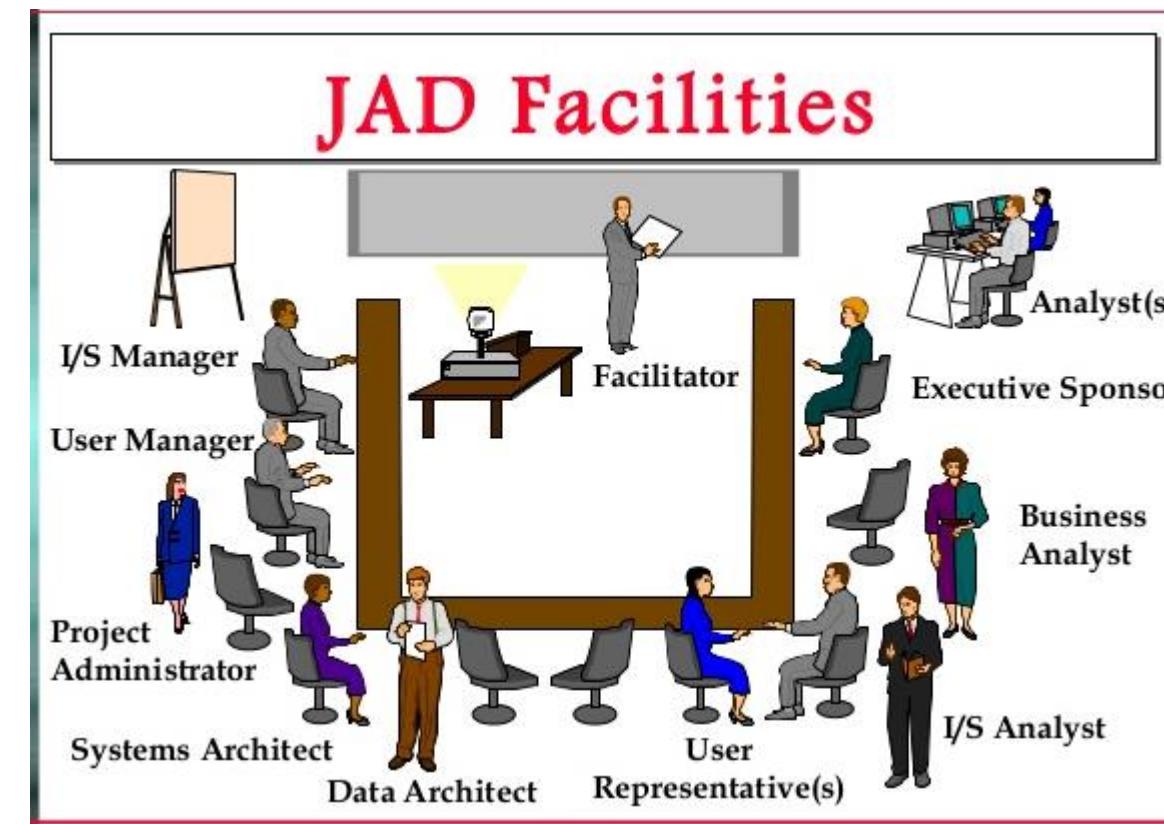


İsterlerin Belirlenmesi Aşamasının Genel Hatları

JAD (Joint Application Design)

JAD metodunda da katılımcılar mevcut. Katılımcı rolleri şu şekildedir;

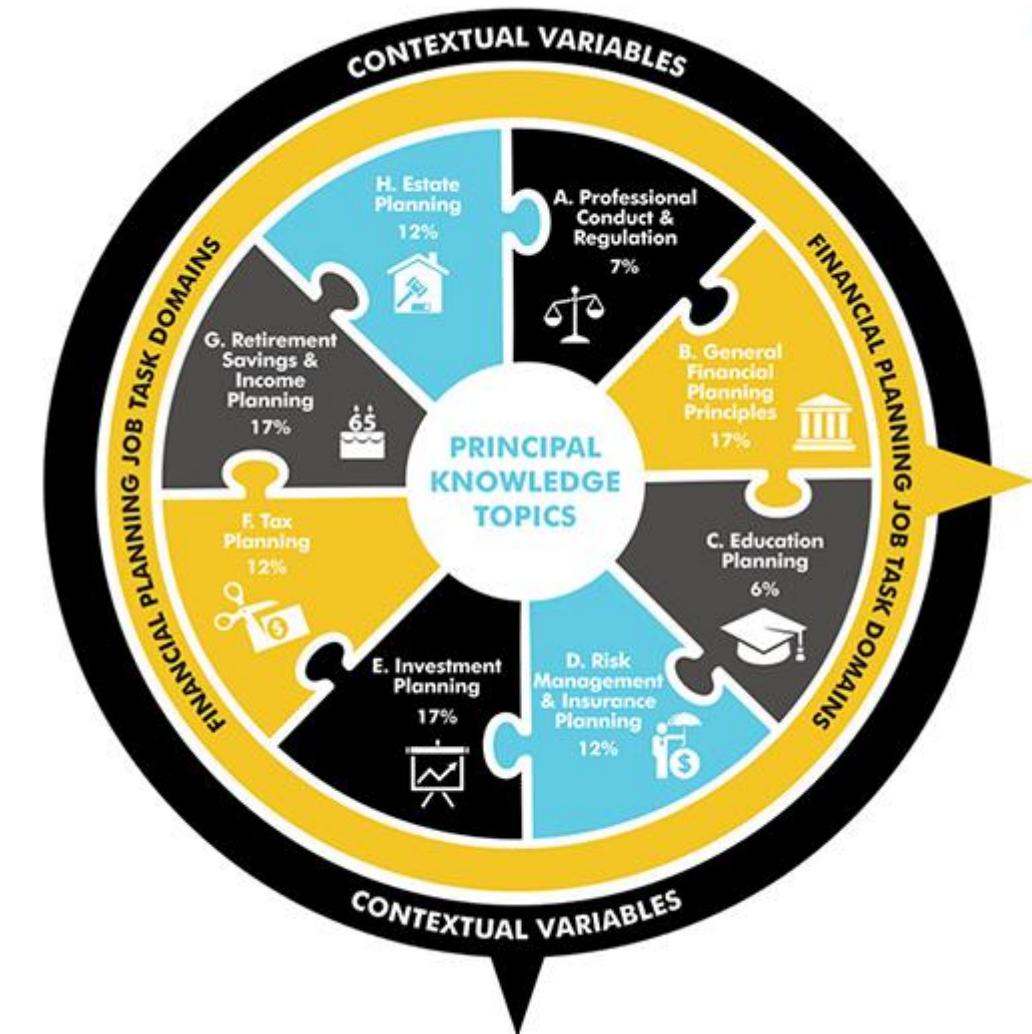
- Executive Sponsor – Proje Yürütme Sponsoru
- JAD Facilitator – Workshop Yöneticisi
- Documentor – Yazıcı (Scribe)
- Business Users – Business Tarafı Kullanıcısı / Experti (Bilir Kişi)
- Technical Support ya da System Expert – Sistem Bilir Kişi
 - (İstenildiği takdirde) Outside Experts – Eğer istenilirse dışarıdan çağırılan danışman/bilir kişi
- Observers – Gözlemciler



İsterlerin Belirlenmesi Aşamasının Genel Hatları

KAD (Knowledge Analysis of Task)

- Sistem geliştiricilerin gözlemlerine dayanarak bilgi edinmeleridir.



İsterlerin Belirlenmesi Aşamasının Genel Hatları

Kullanılabilirlik Testleri (Usability Testing)

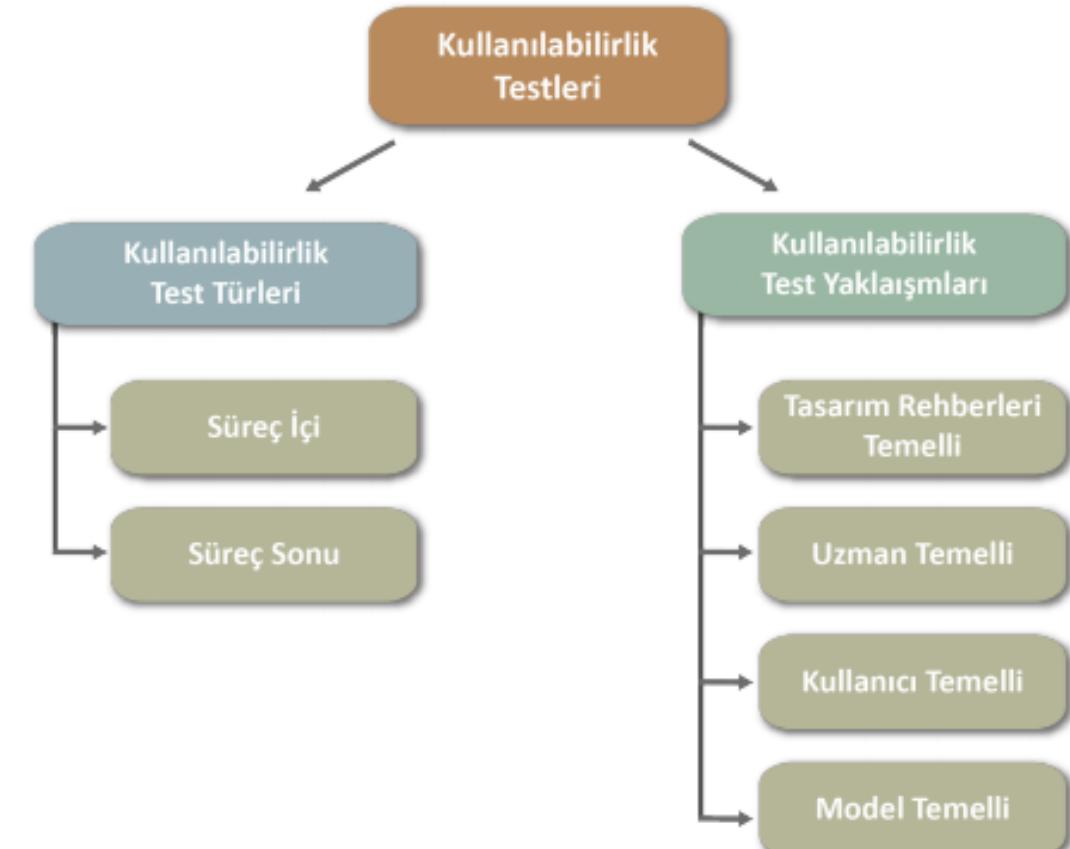
- İsterlerin belirlenmesi modelinin geçerliliğinin çeşitli metodlar ile sınanmasıdır.
- Sistemin kullanıcıların ihtiyaç ve bekentilerini karşılayıp karşılamadığının ve sitenin kullanılabilirliğinin ölçülmesi amacıyla kullanılabilirlik testleri uygulanmaktadır.
- Kullanılabilirlik testleri, testin amacına göre “Tür” ve elde edilen verinin kaynağuna göre “Yaklaşım” olmak üzere iki temel başlık altında toplanmaktadır.



İsterlerin Belirlenmesi Aşamasının Genel Hatları

Kullanılabilirlik Testleri (Usability Testing)

- Tür başlığı, Süreç İçi (Formative) ve Süreç Sonu (Summative) olmak üzere ikiye ayrılmaktadır.
- Yaklaşım başlığı ise, Tasarım Rehberleri Temelli, Uzman Temelli, Kullanıcı Temelli ve Model Temelli olmak üzere dörde ayrılmaktadır.



İsterlerin Belirlenmesi ile İlgili Kavramlar

Bu bölümde isterlerin belirlenmesi ile ilgili temel kavramlar incelenecektir.

- İşlevsel isterler (Functional Requirements)
- İşlevsel olmayan ve sözde isterler (Nonfunctional and Pseudo Requirements)
- İsterlerin doğrulanması (Requirements Validation)
- Greenfield Engineering, Reengineering, Interface Engineering



FONKSİYONEL VE FONKSİYONEL OLMAYAN GEREKSİNİMLER

Yazılım sistemi gereksinimleri genellikle fonksiyonel ve fonksiyonel olmayan gereksinimler olarak sınıflandırılır:

1. Fonksiyonel gereksinimler Bunlar sistemin sunması gereklili servislerin, sistemin belli girdilere nasıl tepki vermesi gerektiğini, belli durumlarda sistemin nasıl davranışması gerektiğini ifadesidir. Fonksiyonel gereksinimler bazı durumlarda sistemin ne yapmaması gerektiğini de açıkça belirtebilir.
2. Fonksiyonel olmayan gereksinimler Bunlar sistemin sunduğu servisler ve fonksiyonlar üstündeki kısıtlamalarıdır. Zaman kısıtlamalarını, geliştirme süreci üstündeki kısıtlamaları ve standartlarla dayatılan kısıtlamaları içerirler. Fonksiyonel olmayan gereksinimler genellikle tek tek sistem olanakları ve servislerinden çok, sistemin bütününe uygulanır.



İsterlerin Belirlenmesi ile İlgili Kavramlar

İşlevsel İsterler (Functional Requirements)

- Sistemin sunacağı hizmetler ile sistemin işlevsel altyapısını tanımlarlar. Sistemin ne yapacağını yapısal ve işlevsel olarak ortaya koyarlar. Geliştirmeden bağımsız çoğunlukla giriş, çıkış arabirimleri, süreçler ile hata yönetimine yönelik gereksinimlerdir.
- Sistem girişindeki izin verme ve yetkilendirme gereksinimleri de bu tiptedir. Sistemin neler yapacağını soyut olarak değil de detaylandırılmış biçimde belirler.
- Fonksiyonel gereksinim analizi fonksiyonel analiz için yüksek seviye fonksiyonlarda kullanılır.



İsterlerin Belirlenmesi ile İlgili Kavramlar

İşlevsel İsterler (Functional Requirements)

İşlevsel isterler sistemin etkileşim içinde olduğu kullanıcı ve diğer sistemlerle olan etkileşiminden kaynaklanan isterlerdir.

- **ÖRNEK:** Bulunulan yere göre zamanı gösteren saat örneğini ele alalım. Bu saat GPS (Global Positioning System) uydularını kullanarak bulunulan konumu belirliyor; sonra da sahip olduğu veri yapıları ile bu konumu zaman dilimine dönüştürüyor.

Aşağıda açıklanan özellikler bu saatin işlevsel isterleridir.

- Saatin kullanıcısı bir zaman diliminden diğerine geçerken saat, otomatik olarak saati ve tarihi ayarlıyor.
- Saatin kullanıcısı bir ayar yapmak zorunda kalmıyor. Bu nedenle bu saatin kullanıcıya yönelik herhangi bir kontrol butonu yoktur.
- Saatin 2 satırlık gösterim menüsü vardır.
- Üst satırda zaman (saat, dakika, saniye, zaman dilimi) görüntüleniyor.
- İkinci satırda da tarih (haftanın günü, gün, ay ve yıl) görüntüleniyor.
- Görüntüleme teknolojisi çok zayıf ışıkta bile kullanıcının saati ve tarihi görebileceği şekilde kullanılmış.

İşlemci, programlama dili, performans, görüntüleme teknolojisi gibi **sistemin gerçeklenmesi ile ilgili olan detaylar** işlevsel isterler grubuna girmezler.

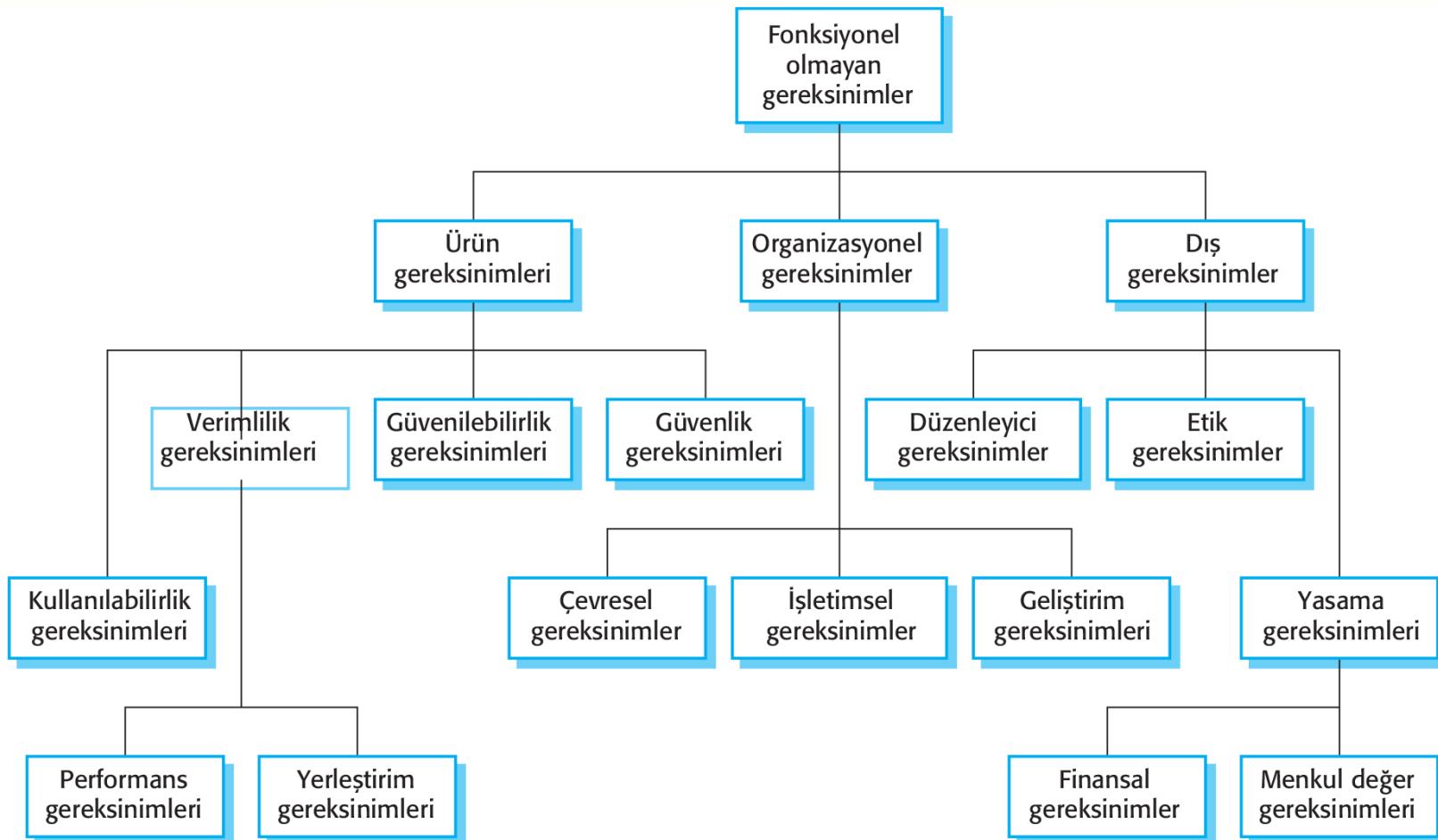


İsterlerin Belirlenmesi ile İlgili Kavramlar

İşlevsel Olmayan ve Sözde İsterler (Nonfunctional and Pseudo Requirements)

- Sistemin daha çok kısıtları ile fiziksel ortam, arayüzler, kullanıcı odaklı olma, güvenlik, güvenilirlik, kalite güvence gibi soyut niteliklerini belirleyen gereksinimlerdir.
- Yazılımlara işlevsellik katmamasına rağmen bu tip gereksinimler özellikle yazılım kalitesi açısından kritik rol oynarlar. Bu gereksinimler yazılımda karşılanmadığı sürece yazılımın kullanılabilirliği yetersiz kalacaktır
- *İşlevsel olmayan isterler kullanıcı tarafından görülen fakat işlevsellikle ilgili olmayan isterlerdir.*
- *Tepki süresi (response time) ve doğruluk derecesi (accuracy) gibi kısıtlar (constraints) bu gruba girer.*
- *İşlevsel olmayan isterler, kullanıcı tarafından görülen fakat işlevsellikle ilgili olmayan isterlerdir.*
- *Sözde isterler, müşteri tarafından sistemin gerçekleştirilmesi (implementation) ile ilgili olarak getirilen kısıtlardır (constraints).*





Şekil 4.3 Fonksiyonel olmayan gereksinim tipleri

ÜRÜN GEREKSİNİMİ

ZihinSaS sistemi bütün klinikler için normal çalışma saatlerinde (Pazartesi-Cuma, 08.30-17.30) erişilebilir olmalıdır.

ORGANİZASYONEL GEREKSİNİM

ZihinSaS sisteminin kullanıcıları kendilerini sağlık yetkilisi kimlik kartlarını kullanarak tanıtmak zorundadır.

DIŞ GEREKSİNİM

Sistem Hstan-03-2006-priv de belirtilen gizliliği hükümlerini yerine getirmelidir.

Şekil 4.4 ZihinSaS
sistemi için fonksiyonel
olmayan gereksinim
örnekleri



Nitelik	Ölçüt
Hız	İşlem/saniye Kullanıcı/vaka cevap süresi Ekran yenilenme zamanı
Büyüklük	Megabayt/ROM yonga sayısı
Kullanım kolaylığı	Eğitim zamanı Yardım ekranlarının sayısı
Güvenilirlik	Hata ortalama zamanı Kullanılamama olasılığı Hata sıklığı Kullanılabilirlik
Dayanıklılık	Hatadan sonra tekrar başlama süresi Hataya neden olan vakaların yüzdesi Hata durumunda verinin bozulması olasılığı
Taşınabilirlik	Hedef bağımlı bildirimlerin oranı Hedef sistemlerin sayısı

Şekil 4.5 Fonksiyonel olmayan gereksinimlerin spesifikasyon ölçütleri



İsterlerin Belirlenmesi ile İlgili Kavramlar

İşlevsel Olmayan ve Sözde İsterler (Nonfunctional and Pseudo Requirements)

- Bir önceki konuda tarif edilen GPS'li saatin, işlevsel olmayan isterlerini ve sözde isterlerini aşağıdaki gibi belirtebiliriz:

İşlevsel Olmayan İsterler

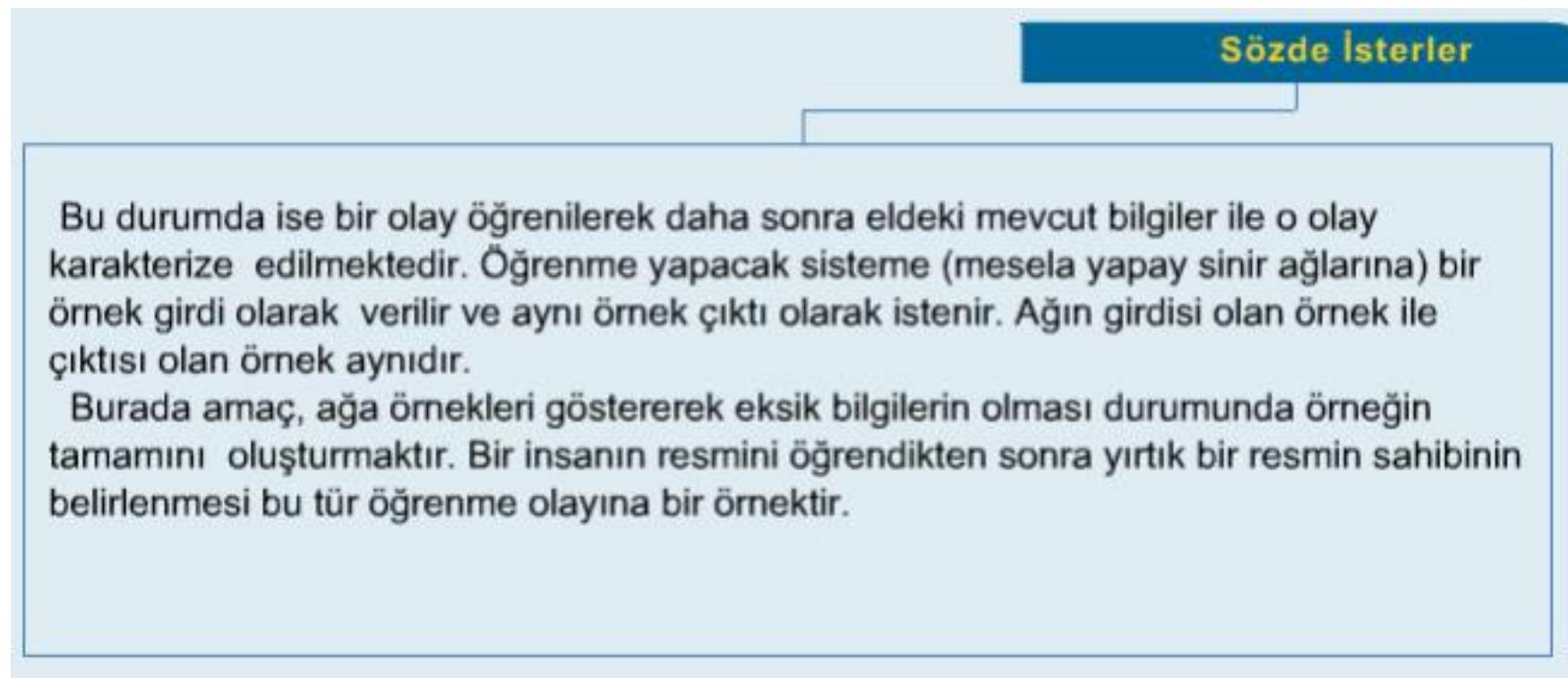
Saat, konumu, GPS uydularını kullanarak belirlediğine göre GPS uydularının maruz olduğu bütün sınırlamalara sahiptir.

- Yaklaşık 100 feet doğruluk derecesi
- Dağlık bölgelerde günün belirli saatlerinde konumu belirleyememe
- Saatin pilinin ömrü 5 yıl ve pil bittikten sonra değiştirilmeye elverişli değil
- Yani pili tükenince saatin de ömrü bitmiş oluyor

İsterlerin Belirlenmesi ile İlgili Kavramlar

İşlevsel Olmayan ve Sözde İsterler (Nonfunctional and Pseudo Requirements)

- Bir önceki konuda tarif edilen GPS'li saatin, işlevsel olmayan isterlerini ve sözde isterlerini aşağıdaki gibi belirtebiliriz:



İsterlerin Belirlenmesi ile İlgili Kavramlar

İsterlerin Doğrulanması (Requirements Validation)

- İsterler belirlendikten sonra müşteri tarafından doğrulanmalıdır. İsterlerin doğrulanması için gerekli kriterler aşağıda özetlenmiştir:

Kriter	Açıklama
Doğruluk (Correctness)	İsterler müşterinin bakış açısını yansıtmalı
Tamlık (Completeness)	Olası bütün senaryolar ve istisnai durumlar ele alınmış olmalı
Tutarlılık (Consistency)	Bir birleri ile çelşen işlevsel veya işlevsel olmayan isterler bulunmamalı
Açıklık (Clarity)	İsterlerde belirsizlik olmamalı
Gerçekcilik (Realism)	Belirlenen isterler gerçekleştirilebilri ve teslim edilebilir olmalıdır.
İzlenebilirlik (Traceability)	Her sistem işlevinden ilgili işlevsel istere erişilebilmeli



İsterlerin Belirlenmesi ile İlgili Kavramlar

Doğruluk (Correctness)

- Gereksinimler, sunulacak olan fonksiyonu doğru olarak tanımlamalıdır.
- İstenilen ile gereksinimde açıklananların farklı olması halinde gereksinim kaynağına bakılmalıdır.
- Gereksinimlerin türüne göre uygun kaynağın belirttiği ifadeler dikkate alınmalıdır.
- Çatımlar olması halinde her iki kaynağı da gereksinimler doğrulatılmalıdır.
- Bu yöntemde yanlış gereksinime neden olan kaynak ortaya çıkacaktır.
- Çoğunlukla üç kullanıcılar ile yöneticilerin gereksinimleri birbiriyle çakışabilmektedir.
- Bunun nedeni de yöneticinin olayların detayına inmeden, yüzeysel bakı açısından gereksinimlere odaklanması, üç kullanıcının da yönetimsel kararlara bağlı gereksinimler hakkında hükmü vermesidir. Örneğin, yönetim tarafından alınmamasına karar verilen haftalık bir raporu, o raporu önceden takip eden kullanıcı zorunlu bir gereksinim olarak sunabilir.



İsterlerin Belirlenmesi ile İlgili Kavramlar

Tamlık (Completeness)

- Olası bütün senaryolar ve istisnai durumlar ele alınmış olmalı
- Gereksinimde karşılaşılabilen tüm koşullar belirtilmiş mi? Müşteriden alınan tüm gereksinimler dokümanda yer alıyor mu?



İsterlerin Belirlenmesi ile İlgili Kavramlar

Tutarlılık (Consistency)

- Bir birleri ile çelişen işlevsel veya işlevsel olmayan isterler bulunmamalı
- Gereksinimler birbirleriyle tutarlı olmalıdır. Birbiriyle çelişen, çatışan, bütünlüğü bozan gereksinimler olmamalıdır.
- Özellikle gereksinim kaynaklarının çeşitliği nedeniyle bir kaynaktan alınan gereksinim diğeriyile çelişebilir.
- Bu durumda gereksinimler arasında izlenebilirlik ve ilişki şemalarının hazırlanması ve tutarsızlıkların tespit edilmesi gereklidir.
- Bu durum gereksinimlerin doğruluunu da azaltacaktır. Doruluk kısmında açıklamalar yapılmıştır



İsterlerin Belirlenmesi ile İlgili Kavramlar

Açıklık (Clarity)

- Gereksinimler tek bir konudaki gereksinimi açıklamalı ve açık, öz, anlaşılır ve basit olmalıdır.
- Gereksinimler ve gereksinim dokümanlarının edebi içeriklere sahip olmasına gerek yoktur.
- Ağdalı ve anlaşılması güç ifadelerin kullanılması halinde gereksinimin asıl amacından çıkılabilir.
- Basit ve bağlantı ekleri içermeyen, muğlak olmayan, ihtimallere yer vermeyen, tartışılmayacak kadar açık ve net ifadeler ile iyi bir gereksinim yazılır.
- Gereksinimi okuyan kişiler gereksinim üzerinde yorum yapmasına gerek duymayacak biçimde muğlak olmayan, açık ve net olmalıdır.
- Konuşma dilinde yazılan gereksinimler belirsiz olmaya yatkındır. Belirsizliği azaltmak için şu kelimelerden kaçınılmalıdır: Kolay, basit, sade, hızlı, verimli, birkaç, maksimum, minimum.



İsterlerin Belirlenmesi ile İlgili Kavramlar

Gerçekcilik (Realizm) ve Ulaşılabilirlik

- Belirlenen isterler gerçekleşebilir ve teslim edilebilir olmalıdır.
- Gereksinimler, sistemin kısıtları kapsamında ulaşılabilir ifadeler olmalıdır.
- Projenin bütçe, zaman ve insan kaynağı ile gerçekleştirilemeyecek gereksinimler erişilebilir değildir.
- Gereksinimler, projenin kısıtlarını zorlamayacak şekilde ya yeniden düzenlenmelidir ya da tamamen dokümanlardan çıkarılmalıdır.
- Gereksinimin erişilebilir olup olmadığını anlayabilmek için teknik fizibilite çalışması yapılmalıdır.



İsterlerin Belirlenmesi ile İlgili Kavramlar

Gerçekcilik (Realizm) ve Ulaşılabilirlik

- Teknik olurlüğün olması bile projenin zaman ve bütçe kısıtları nedeniyle ilgili gereksimin erişilebilir olmasını sağlayamaz.
- Kullanıcılara izin verilirse yazılımın gelecektен haber vermesini bile isteyebilirler.
- Her gereksiminin ek kaynak ve maliyet olarak düşünülmesi ve proje planındaki sınırlara uyulması gereklidir.
- Teknik olurluğu sağlanan ama kaynak açısından kabul edilemeyen gereksimlerin zorunluluğu ortaya çıkarsa, müşterinin bilgisi dahilinde proje planının revize edilmesi ve zaman planının yeniden yapılması gereklidir..



İsterlerin Belirlenmesi ile İlgili Kavramlar

Izlenebilirlik (Traceability)

- Her sistem işlevinden ilgili işlevsel istere erişilebilmeli
- Gereksinimin kaynağı belli mi? Gereksinimler arasında ilişki kurulmu mu? Kurulmu ilişkilerde iç içe geçme durumu var mı? Döngüsel izlenirlik olu mu?



İsterlerin Belirlenmesi ile İlgili Kavramlar

Greenfiled Engineering, Reengineering, Interface Engineering

- İsterlerin belirlenmesi aşamasını farklı kılan tipler aşağıda listelenmiştir.:

Tip	Açıklama
Greenfiled Engineering	Mevcut sistem yoktur. Sistem başttan geliştirilir. Geliştirme kullanıcıların ihtiyaçlarından kaynaklanır. İsterler kullanıcılar ve müşteriler tarafından belirlenir.
Reengineering,	Mevcut olan sistemi yeni bir teknoloji ile yeniden tasarlamak ve gerçekleştirmek.
Interface (Arayüz) Engineering	Mevcut bir sistemin yeni bir ortamda kullanılması. Yeni bir teknolojinin veya yeni ihtiyaçların ortaya çıkması bu mühendisliği gerekli kılar

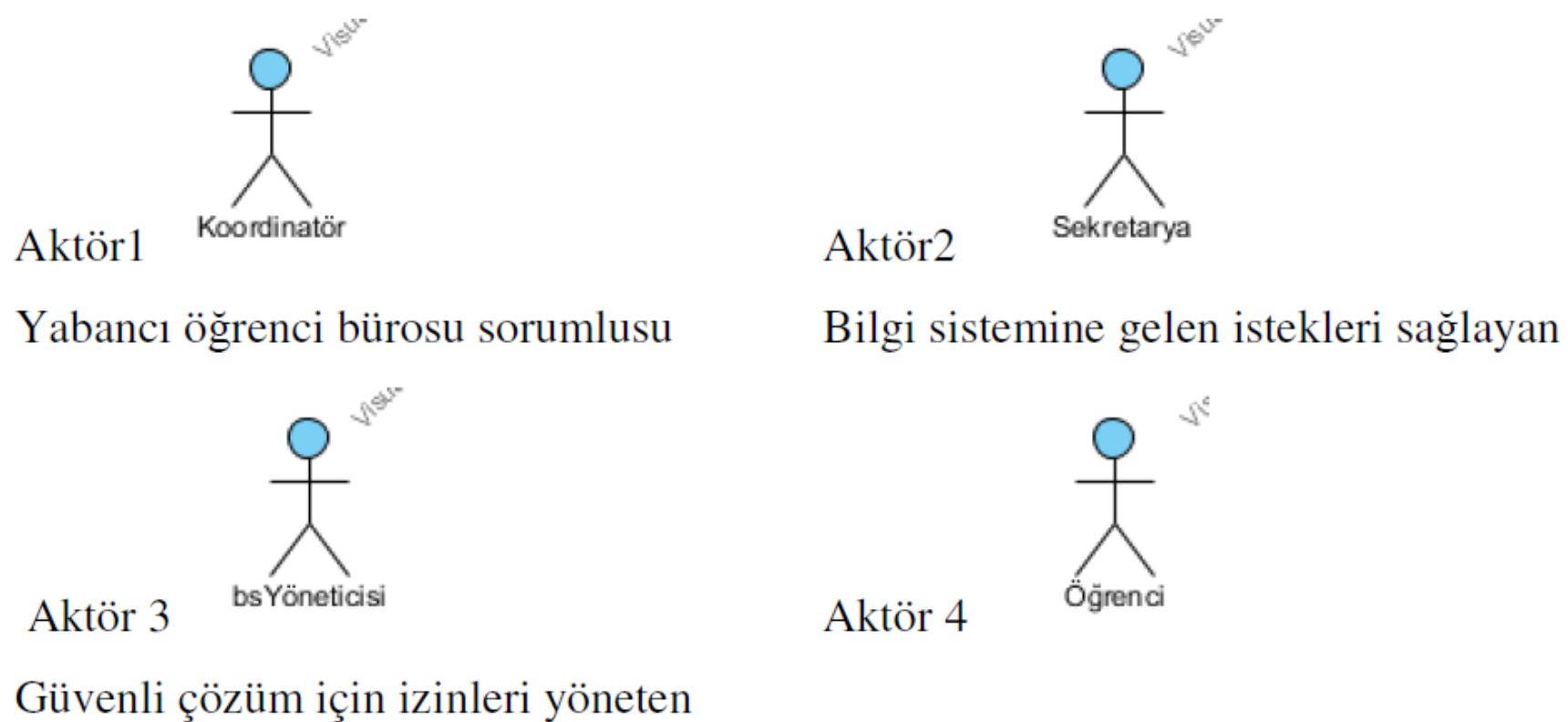


Örnek Problem

- “Üniversiteye devam edecek yabancı bir öğrencinin ülkede yaşayabilmek için izin alması gerekmektedir. Bunun için de üniversiteye kaydını yaptırdığına iliskin bir belgeye ihtiyacı vardır; ayrıca kişi o ülkede yaşamak için yeterli miktarda geliri olduğunu da kanıtlamalıdır. Hazırlanacak bu belge program koordinatörü tarafından onaylanmalıdır. Koordinatör üniversite bilgi sisteminde kayıtlı olan kişisel bilgilerin alınması için sekretaryayı yetkilendirir. Sekretarya öğrencinin bilgilerini alarak belgeyi istenildiği şekilde hazırlar ve program koordinatörüne öderilerek onaylatır. Bilgi sistemi yöneticisi sistemin kısıtlanmış bilgilerine de erişerek tüm otomatik (authorisations) düzenlemeleri yönetir.”

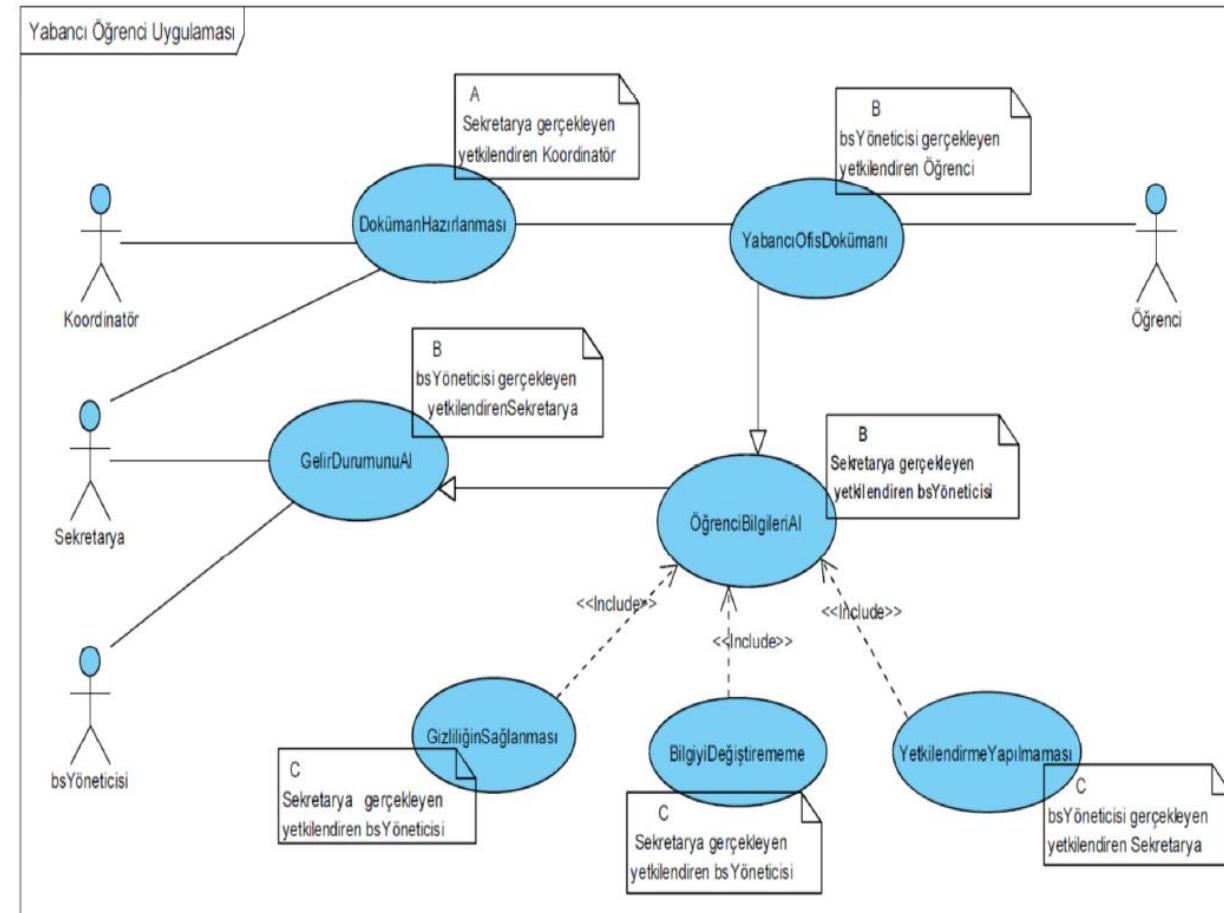


Örnek Problem



Şekil 3: Yabancı Öğrenci Uygulaması Çözümünde Kullanılan Aktörler

Örnek Problem



Şekil 4: Yabancı Öğrenci Uygulamasına ait Güvenilir Gereksinimlerin Use Case Çizeneği

3.Hafta

İsterlerin Belirlenmesi Aktiviteleri



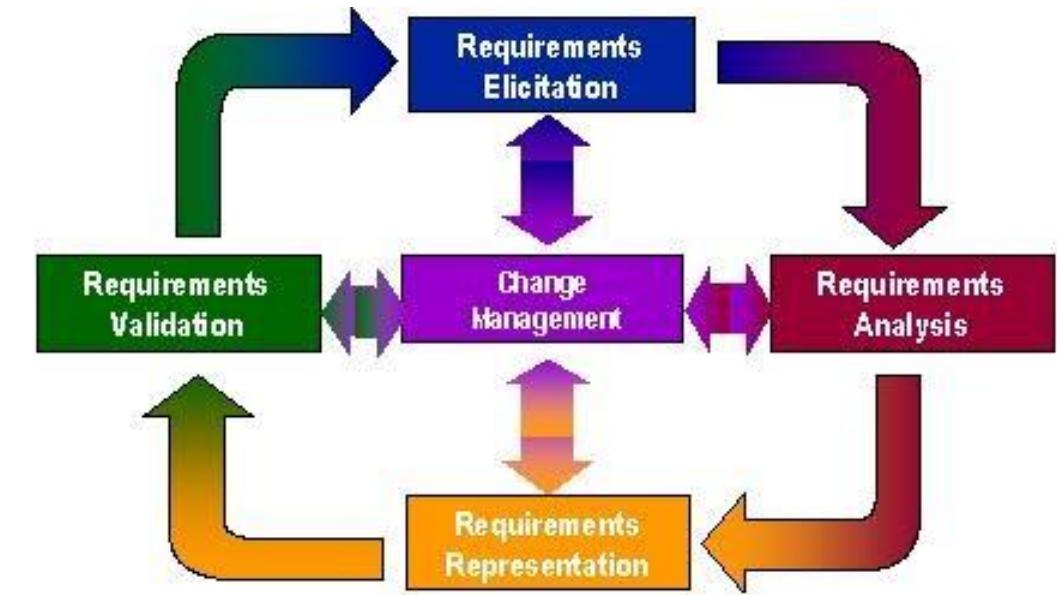
Yazılım İsterleri Çözümlemesi

- Yazılım geliştirme sürecinin başarısı için gereksinim analizi (requirements analysis) ya da yazılım isterleri çözümlemenin (software requirements analysis) doğru yapılması son derece önemlidir.
- Bir yazılım ne kadar iyi tasarılanırsa ya da geliştirilirse geliştirilsin müşteri (ya da son kullanıcı) ihtiyaçlarını karşılamıyorsa başarılı sayılamaz.
- **Yazılım isterlerinin çözümlenmesi aşamasında müşterinin yazılımdan ne beklediği ayrıntılı olarak belirlenir, gereksinimler açığa çıkarılır, yazılım isterleri modellenir ve tanımlanarak tasarım aşamasına geçilir.**



İSTERLERİN BELİRLENMESİ (REQUIREMENTS ELICITATION)

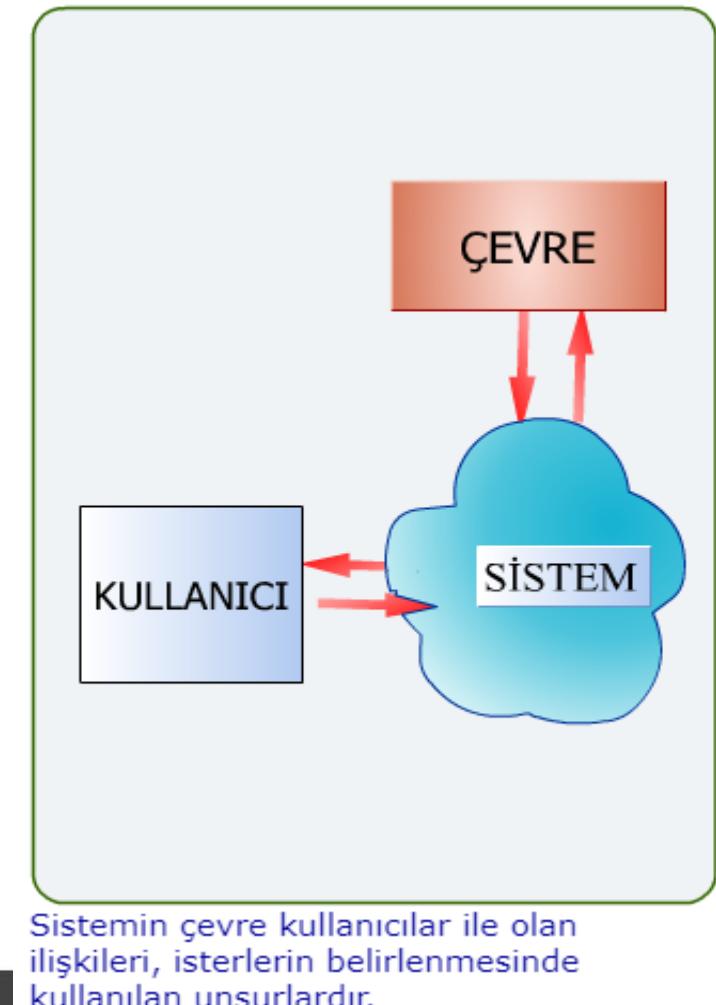
- Bir sistemin, müşterileri tarafından kabul edilebilmesi için, sağlaması gereken **kısıtlara (constraints)** o sistemin **isterleri (requirements)** denir.
- Sistem geliştiricilerinin anlayabileceği ve yorumlayabileceği **isterler**, analiz modeli geliştirilmeden önce belirlenir ve müşterinin rahatlıkla anlayabileceği bir şekilde sistem spesifikasyonu oluşturulur.
- İsterlerin belirlenmesi aşaması oldukça zordur, çünkü bu aşama farklı niteliklere sahip olan müşterilerin, kullanıcıların ve sistem geliştiricilerin ortak çalışmasını gerektirir.



İsterlerin Belirlenmesi Aşamasının Genel Hatları

İsterlerin belirlenmesi aşamasında izlenecek adımlar:

- Aktörlerin Belirlenmesi (Identifying Actors)
- Senaryoların Belirlenmesi (Identifying Scenarios)
- Kullanım Durumları (Use Case)'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use Cases)
- Aktörler ve Kullanım Durumları Arasındaki ilişkilerin Belirlenmesi (Identifying Relationships Among Actors and Use Cases)
- Sistemdeki Objelerin Ön Belirlenmesi (Identifying Initial Analysis Objects)
- İşlevsel Olmayan İsterlerin Belirlenmesi (Identifying NonFunctional Requirements)



1. Aktörlerin Belirlenmesi (Identifying Actors)

- **Aktörler sistemle etkileşim ve bilgi alışverişi içinde bulunan kişi ya da başka sistemlerdir.**
- Gereksinimlerin ortaya çıkarılmasının ilk adımı, aktörlerin tanımlanmasıdır.
- Bu, hem sistemin sınırlarını tanımlamaya hem de geliştiricilerin sistemi dikkate almaları gereken tüm perspektifleri bulmaya hizmet eder.
- Sistem mevcut bir organizasyona (şirket gibi) yerleştirildiğinde, çoğu aktör genellikle sistem geliştirilmeden önce var olurlar: Organizasyondaki rollere karşılık gelirler.,



1. Aktörlerin Belirlenmesi (Identifying Actors)

- Aktör tanımlamanın kimliğinin ilk aşamalarında, aktörleri nesnelere ayırmak zordur. Örneğin, bir veritabanı alt sistemi bazen bir aktör olabilirken, diğer durumlarda sistemin bir parçası olabilir.
- Sistem sınırı tanımlandıktan sonra, aktörler ve bu tür sistem bileşenlerini nesne veya alt sistem olarak ayıran bir sorun olmadığını unutmayın.
- Aktörler sistem sınırının dışındadır;
- Alt sistemler ve nesneler sistem sınırının içinde bulunur; onlar iç. Böylece, geliştirilecek sistemi kullanan herhangi bir harici yazılım sistemi bir aktördür.



1. Aktörlerin Belirlenmesi (Identifying Actors)

- Aktörleri belirlemek için sistem geliştiricileri aşağıdaki soruları sorabilirler:

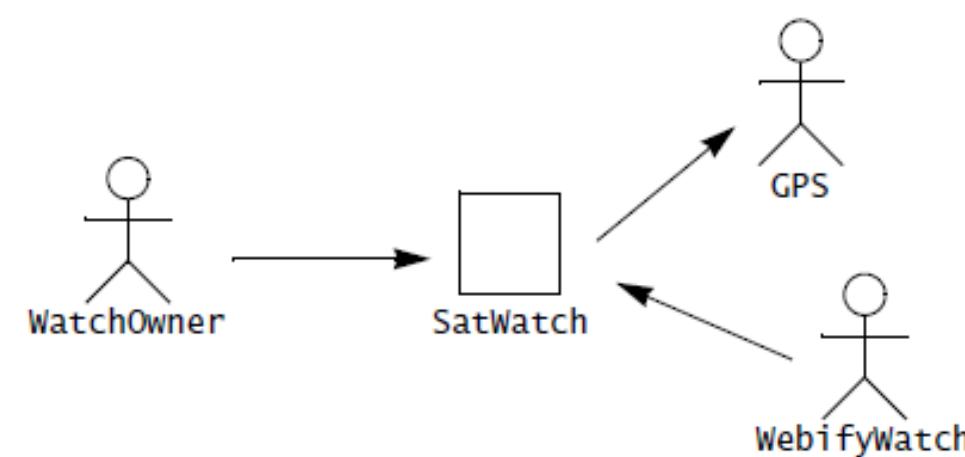
- Sistem hangi kullanıcı gruplarının işlerini yapabilmesini sağlamaktadır?
- Sistemin temel fonksiyonlarını hangi kullanıcı grupları yürütmektedir?
- Hangi kullanıcı grupları sistemin yönetimi ve bakımı gibi ikincil fonksiyonları yürütmektedir?
- Sistem başka bir yazılım veya donanım sistemi ile etkileşim içinde bulunacak mı?

- Aktörler belirlendikten sonra her aktörün sistemle gerçekleştirebileceği işlevler belirlenmeli. Bunu Senaryolar ve Senaryoların biçimlendirilmiş şekli olan Kullanım Durumları vasıtasiyla yapıyoruz.
- Aktörler belirlendikten sonra, gereksinim belirleme etkinliğindeki bir sonraki adım, her aktörün erişebileceğini işlevselligi belirlemektir. Bu bilgiler senaryolar kullanılarak çıkarılabilir ve kullanım durumları kullanılarak resmileştirilebilir.



1. Aktörlerin Belirlenmesi (Identifying Actors)

- SatWatch örneğinde, saat sahibi, GPS uyduları ve WebifyWatch seri cihazı aktörlerdir
- Hepsi SatWatch ile bilgi alışverişi yapıyorlar.
- Bununla birlikte, hepsinin SatWatch ile belirli etkileşimlere sahip olduğunu unutmayın: saat sahibi giyer ve saatine bakar; saat, GPS uydularından gelen sinyali izler; WebifyWatch, yeni verileri saatin içine indirir. Aktörler işlevsellik sınıflarını tanımlar.



1. Aktörlerin Belirlenmesi (Identifying Actors)

ÖRNEK: KAZA TAKİP VE MÜDAHALE (KTM)

- KTM (Kaza Takip ve Müdahale) sistemi, kaza takibi ve müdahalelesini sağlayan bir dağıtılmış bilişim sistemidir (distributed information system).

KTM sisteminde yer alan kişiler ve sistemin işleyışı şöyledir;

Bölge sorumlusu bir polis memuru veya bir itfaiye memuru olabilir.

Görev dağıticisi gelen kaza raporlarını değerlendirmek ve kaza mahalline polis veya itfaiye gibi kaynakları sevk etmek ile görevlidir.

Bölge sorumluları kaza vakalarını iletmek ve görev dağıticisinin kendilerini sevk ettiği kaza mahalline gidip gerekli müdahaleyi yapmakla görevlidirler.

Bölge sorumluları KTM sistemine mobil PDA (personal digital assistant)'ları ile, görev dağıticileri ise sisteme iş istasyonu (workstation) ile erişirler.



Tekrar

1. Aktörlerin Belirlenmesi (Identifying Actors)

ÖRNEK: KAZA TAKİP VE MÜDAHALE (KTM)

- KTM sistemi kaza takibini, görev ve kaynak dağıtımını ve planlanmasını sağlamaktadır.
- KTM Sistemi için Aktörlerin belirlenmesi:
- Aktörlerin belirlenmesi için sorulan sorular, KTM sistemi için birçok potansiyel aktör yaratmasına yol açmaktadır: itfaiyeci, polis memuru, görev dağıtıcısı, sistem yöneticisi, sistemin yatırımcısı, vali gibi.
- Bu potansiyel aktörler incelendiğinde itfaiyeci ve polis memurunun aynı arayüzü kullanarak sisteme erişliğini görüyoruz, dolayısıyla ikisini Bölge Sorumlusu adı altında bir aktör ile tanımlıyoruz. Görev Dağıtıcısı ise ikinci aktörümüz.
- Vali, sistem yöneticisi ve sistemin yatırımcısının sistemle doğrudan bir etkileşimi yok, dolayısıyla onları aktör olarak tanımlamıyoruz.



2. Senaryoların Belirlenmesi (Identifying Scenarios)

- Senaryo, bir aktörün sistemle etkileşim içindeyken adım adım yaptıklarının ve gözlemlediklerinin anlatımıdır. Sistem, sadece bir aktörün bakış açısıyla ele alınır ve kullanıcıların rahatça anlayabileceği bir dilde yazılır.
- Bir senaryo, “bilgisayar sistemlerini ve uygulamalarını kullanmaya çalışıkça insanların neler yaptıklarını ve deneyimlediklerini anlatır” [Carroll, 1995].



2. Senaryoların Belirlenmesi (Identifying Scenarios)

- Bir senaryo, sistemin tek bir özelliğinin tek bir aktör açısından somut, odaklı, gayri resmi bir tanımlamasıdır.
- Senaryolar, belirli durumlara ve somut olaylara (tam ve genel açıklamaların aksine) odaklandıklarından kullanım durumlarını değiştiremez (ve amaçlamazlar). Bununla birlikte, senaryolar, kullanıcı ve müşteriler için anlaşılabilir bir araç sağlayarak gereksinimlerin ortaya çıkarılmasını sağlar.



2. Senaryoların Belirlenmesi (Identifying Scenarios)

- ReportEmergency (use case) kullanım durumu için Depo Yangını «warehouseOnFire (sistem)» acil durum senaryosu örneği

Scenario name: warehouseOnFire

Participating actors Instances : bob, alice:FieldOfficer
john:Dispatcher

1. Bob, devriye arabasında ana caddeyi aşağı doğru sürerken, bir depodan çıkan dumanı fark eder. Ortağı Alice, FRIEND dizüstü bilgisayarından “Acil Durum Raporu” işlevini etkinleştirir.
2. Alice, binanın adresini, bulunduğu yerin kısa bir açıklamasını (yani, kuzeybatı köşesi) ve bir acil durum seviyesini girer. Bir yanın ünitesine ek olarak, bu alanın göreceli olarak yoğun olduğu göz önüne alındığında, olay yerinde birkaç sağlık görevlisi birimi talep ediyor. Girişini onaylıyor ve bir onay bekliyor.
3. Dispatcher John, iş istasyonuna ait bir bip sesiyle acil durum hakkında uyarılır. Alice tarafından sunulan bilgileri gözden geçirir ve raporu kabul eder. Olay yerine bir yanın birimi ve iki paraşüt birimi tahsis ederek tahmini varış saatini (ETA) Alice'e gönderir.
4. Alice onay ve ETA'yı alı.

2. Senaryoların Belirlenmesi (Identifying Scenarios)

- Aşağıda bazı senaryo türleri verilmiştir:

Senaryo Türleri	Açıklama
As –Is Senaryoları (As Is Scenarios)	Şu anki mevcut durumu tarif etmek için kullanılır. Genellikle mühendislik projelerinde kullanılır. Sistem, kullanıcılar tarafından açıklanır. Mevcut sistem yeniden geliştirilirken uygulanır (re engineering)
Hayali Senaryolar (Visionary scenario)	Gelecekteki sistemi açıklar. Genellikle greenfield mühendisliği ve yeniden yapılandırma projelerinde kullanılır. Kullanıcı ve geliştirici tek başına oluşturamaz. Vizyoner senaryolar, geliştiricilerin modelleme alanında, gelecek sistemle ilgili fikirlerini geliştirdikçe ve kullanıcıların gereksinimlerini karşılayacak bir iletişim aracı olarak hem bir nokta olarak kullanılır. Vizyoner senaryolar ucuz bir prototip olarak görülebilir.
Değerlendirme senaryosu(Evaluation scenario)	Sistemin değerlendirileceği kullanıcı görevlerini açıklar. Kullanıcılar ve geliştiriciler tarafından değerlendirme senaryolarının işbirlikçi gelişimi de, bu senaryolar tarafından test edilen işlevsellik tanımını geliştirir.
Eğitim senaryosu(Training scenario)	Bir acemi kullanıcıyı bir sistemin kullanımı için sistem üzerinden adım adım yönlendiren talimatların bir açıklaması. Örnek: Tic Tac Toe nasıl oynanır.

2. Senaryoların Belirlenmesi (Identifying Scenarios)

- Senaryoları belirlemek için aşağıdaki sorular sorulabilir.

- Aktörün sistemin gerçekleştirmesini istediği işlevler nelerdir?
- Aktör hangi bilgiye erişiyor? Bu bilgi kimin tarafından oluşturuluyor? Bu bilgi değiştirilebilir veya silinebilir mi? Kim tarafından?
- Aktör hangi dış etkenlerden sistemi haberdar etmeli? Hangi sıklıkta? Ne zaman?
- Aktör hangi durumlarda sistem tarafından haberdar edilmeli? Hangi sıklıkta?



Tekrar

- Senaryo ve aktörler sistem geliştiricilerinin sistemin uygulama alanını, sağlaması gereken işlevleri ve sistemin sınırlarını anlamaları ve tanımlamaları için belirlenir. Senaryo ve aktörler belirlendikten sonra bunlar biçimlendirilip Kullanım Durumları oluşturulur.

3. Kullanım Durumları'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use-Cases)

- Aktörler ve Senaryolar belirlendikten sonra bu bilgiler biçimlendirilerek **Kullanım Durumları (Use Cases)** oluşturulur.
- Bir senaryo, kullanım durumunun bir örneğidir; Yani, bir kullanım durumu belirli bir işlevsellik için olası tüm senaryoları belirler.
- Sistemin belli bir işlevi ile ilgili olan bütün Senaryolar, soyutlama yapılarak (**abstraction**) bir **Kullanım Durumu**'ş eklinde ifade edilir.
- Senaryo, bir Kullanım Durumu'nun somutlaşan bir örneğidir. Bir Kullanım Durumu oluşturulurken önce adı belirlenir.
- Sonra rol alan aktörler belirlenir. Bir kullanım durumu bir aktör tarafından başlatılır.
- Bu Kullanım Durumu'nu tetikleyen durum belirlenir. Olayların akışı üzerinde durulur. Kullanım Durumu'nu sonlandıran durum incelenir. İstisna durumlar saptanır ve özel isterler, yani işlevsel olmayan isterler ve kısıtlar belirlenir.



3. Kullanım Durumları'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use-Cases)

- Kullanım Durumları ve Senaryolar aşağıdaki gibi belirlenebilir:

- Önce bir Senaryo belirlenir ve kullanıcının tercih ettiği tarzda olup olmadığı sorulanır.
- Sonra birçok, çok detaylı olmayan Senaryo belirlenir ve sistemin sınırları tespit edilmeye çalışılır.
- Anlamayı kolaylaştırmak için Kullanım Durumları (Use Case) ve Kullanıcı Arayüzleri (Interface) oluşturularak görsel olarak müşteri ve kullanıcıya sunulur.
- Kullanıcının görevleri, gözlemleyerek ya da anketler doldurularak tespit edilmeye çalışılır.



Tekrar

3. Kullanım Durumları'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use-Cases)

ÖRNEK: KAZA TAKİP VE MÜDAHALE (KTM) DEPO YANGINI SENARYOSU

<i>Use case name</i>	ReportEmergency
<i>Participating actors</i>	Initiated by FieldOfficer, Communicates with Dispatcher
<i>Giriş şartı</i>	FieldOfficer, FRIEND olarak oturum açarak KTM sisteminin Report ReportEmergency işlevini harekete geçirir.
<i>Olay Akışı</i>	<ol style="list-style-type: none">1. FieldOfficer, terminalinin “Acil Durum Raporu” işlevini etkinleştirir.2. FRIEND, FieldOfficer'a bir form sunarak yanıt verir.3. FieldOfficer, acil durum seviyesini seçerek formu doldurur, Durum ve durumun kısa açıklaması. FieldOfficer da acil duruma olası yanıtları açıklar. Form tamamlandıktan sonra FieldOfficer formu gönderir.4. FRIEND formu alır ve Dispatcher'a bildirir.5. Dispatcher gönderilen bilgileri gözden geçirir ve OpenIncident kullanım durumunu çağrırmak suretiyle veritabanında bir Olay oluşturur. Dispatcher bir cevap sefer ve raporu kabul eder.6. FRIEND, onaylamayı ve FieldOfficer'a seçilen yanıt gösterir.



3. Kullanım Durumları'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use-Cases)

ÖRNEK: KAZA TAKİP VE MÜDAHALE (KTM) DEPO YANGINI SENARYOSU

<i>Use case name</i>	ReportEmergency
<i>Participating actors</i>	Initiated by FieldOfficer, Communicates with Dispatcher
<i>Çıkış şartı</i>	FieldOfficer, Dispatcher tarafından gönderilen onayı ve izlenecek planı kabul eder
<i>Kalite Gereksinimleri</i>	<ul style="list-style-type: none">• FieldOfficer'ın raporu 30 saniye içinde onaylanır.• Seçilen cevap, "Dispatcher" tarafından gönderildikten sonra en geç 30 saniyede ulaştırılır.
<i>Istisnalar</i>	Bölge sorumlusu, merkez ile arasındaki bağlantı kesilirse haberdar edilir., Görev dağıticisi, herhangi bir bölge sorumlusu ve merkez arasındaki bağlantı kesilirse anında haberdar edilir.



3. Kullanım Durumları'nın Belirlenmesi ve Detaylandırılması (Identifying and Refining Use-Cases)

- Sistemin işlevsellliğini tanımlamak için senaryoların kullanımı ve kullanım durumları, kullanıcı tarafından geliştirmenin erken aşamasında onaylanan gereksinimleri oluşturmayı amaçlamaktadır.
- Sistemin tasarıımı ve uygulaması başladığında, gereksinim şartnamesini değiştirmenin ve yeni öngörülemeyen Gereksinim eklemenin maliyeti artar..



4. Aktörler ve Kullanım Durumları Arasındaki İlişkilerin Belirlenmesi (Identifying Relationships Among Actors and Use Cases)

- Orta ölçekli sistemlerde bile birçok kullanım durumu vardır.
- Oyuncular ve kullanım durumları arasındaki ilişkiler, geliştiricilerin ve kullanıcıların, modelin karmaşıklığını azaltmasına ve anlaşılabilirliğini artırmasına olanak tanır.
- Sistemi işlevsellik katmanlarında tanımlamak için aktörler ve kullanım durumları arasındaki iletişim ilişkileri kullanılır.

- Aktörler ve Kullanım Durumları arasındaki iletişim ilişkisi
(Communication Relationship)
- Kullanım Durumları arasındaki genişletme ilişkisi (Extend Relationship)
- Kullanım Durumları arasındaki içerme ilişkisi (Include Relationship)



Tekrar

- Olağanüstü ve ortak olay akışlarını ayırmak için ilişkileri genişletiyoruz. Kullanım durumları arasında artıklığı azaltmak için ilişkiler kullanıyoruz.

4.1. İletişim İlişkisi

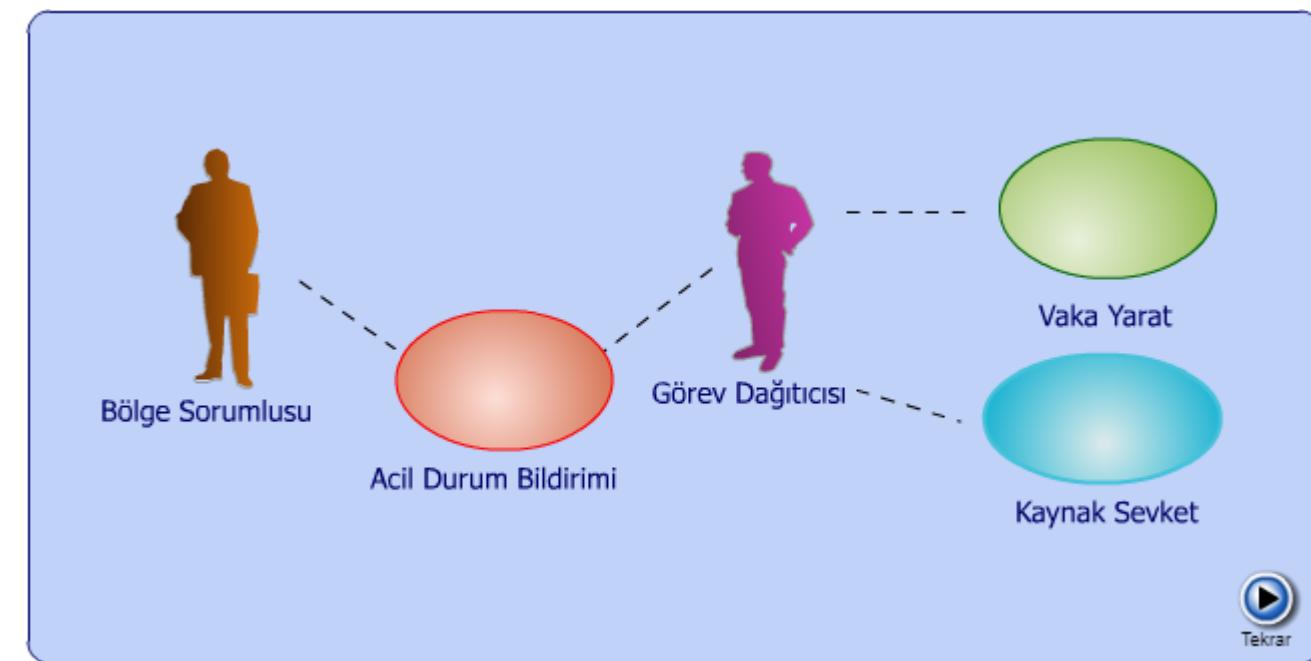
- Aktörler ve kullanım durumları arasındaki iletişim ilişkileri, kullanım durumu sırasında bilgi akışını temsil eder.
- Kullanım durumunu başlatan aktör, kullanım durumunun iletiliği diğer aktörlerden ayırt edilmelidir.
- Hangi aktörün belirli bir kullanım durumunu yerine getirebileceğini belirterek, hangi aktörlerin kullanım durumunu çağrırmayacağını açıkça belirtmek isteriz.
- Benzer şekilde, hangi aktörlerin belirli bir kullanım durumu ile iletişim kurduğunu belirterek, hangi aktörlerin belirli bilgilere erişebileceğini ve hangilerinin mümkün olamayacağını belirleriz.
- Bu nedenle, aktörler ve kullanım durumları arasındaki başlatma ve iletişim ilişkilerini belgeleyerek, sistem için erişim kontrolünü kaba düzeyde belirliyoruz.



4. Aktörler ve Kullanım Durumları Arasındaki İlişkilerin Belirlenmesi (Identifying Relationships Among Actors and Use Cases)

4.1. İletişim İlişkisi

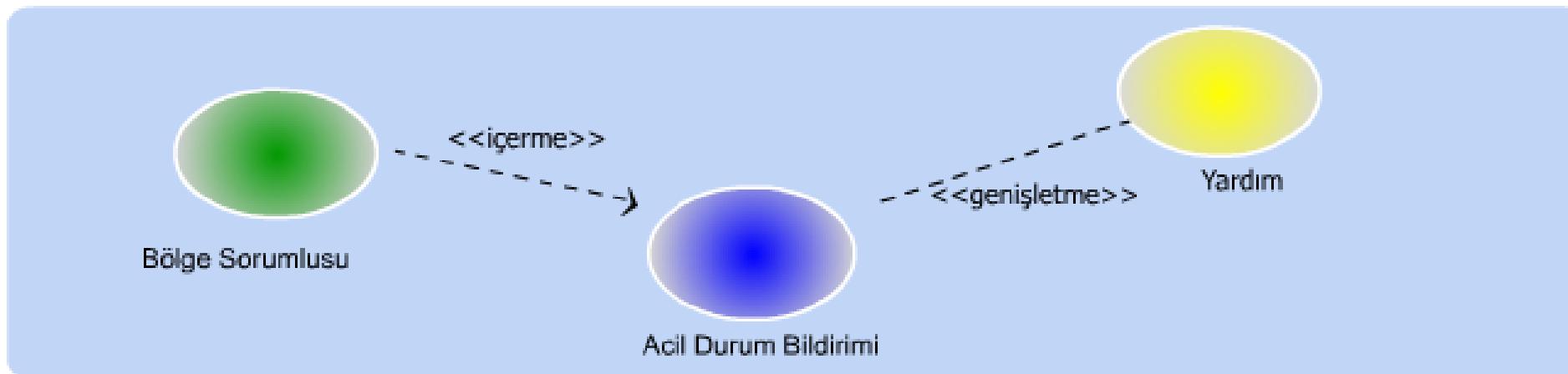
- KTM sisteminde aktörler ve Kullanım Durumları arasındaki iletişim ilişkisini gösteriyor. Şekilden de görüldüğü gibi, Bölge Sorumlusu AcilDurumBildirimi kullanım durumunu, Görev Dağıtıcısı ise VakaYarat ve KaynakSevket kullanım durumlarını başlatıyor. Bölge sorumlusu doğrudan doğruya kendisi vakayı veri tabanına kaydedip (VakaYarat kullanım Durumu) gerekli kaynakları vaka için ayıramıyor (KaynakSevket kullanım Durumu).



Şekil 1. KTM sisteminde aktörler ve Kullanım Durumları arasındaki iletişim ilişkisi

4.2. Genişletme İlişkisi

- Bir Kullanım Durumu, başka bir Kullanım Durumu'nun özelliklerini belli koşullarda barındırıiyorsa buna genişletilmiş Kullanım Durumu denir.
- Örneğimizde, bölge sorumlusu AcilDurumBildirimi yaparken bir sorunla karşılaşrsa yardım almak isteyebilir. Bunun için Yardım Kullanım Durumu yardım sağlama işlevi ile birlikte AcilDurumBildirimi Kullanım Durumunu genişletir. Bu genişletme ilişkisini Şekil 2'de görebilirsiniz.

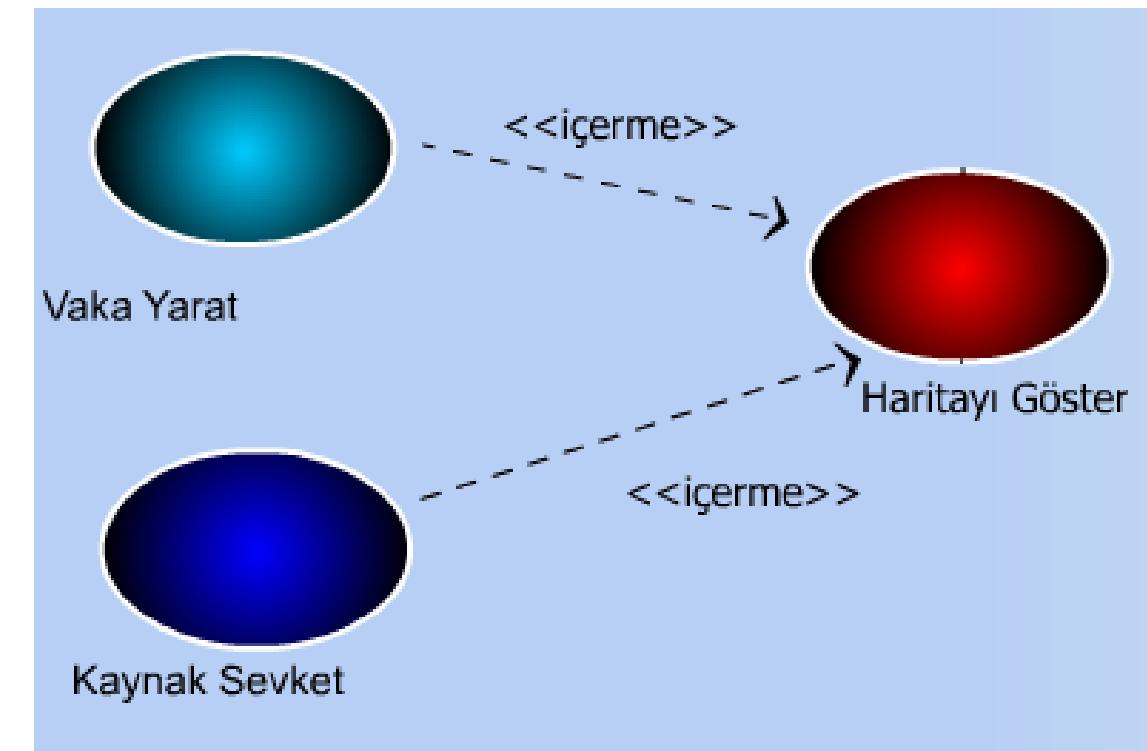


Şekil 2. Genişletme ilişkisine örnek. KTM sisteminde Yardım Kullanım Durumu AcilDurumBildirimi Kullanım Durumu'nu genişletir.

4. Aktörler ve Kullanım Durumları Arasındaki İlişkilerin Belirlenmesi (Identifying Relationships Among Actors and Use Cases)

4. 3. İçerme İlişkisi

- Kullanım Durumları arasındaki artıklıklar (redundancies) içерme ilişkileri ile ortadan kaldırılabilir.
- Örneğimizde, hem vaka ile ilgili bilgiler veri tabanına girilirken, vakaya yakın olan yerleri tespit etmek için bölgenin haritası görüntülenmeli, hem de kaynakları sevketmek için vaka mahaline yakın olan kaynakları tespit etmek için harita görüntülenmeli. Bu durumda hem VakaYarat hem de KaynakSevket Kullanım Durumları HaritayıGöster Kullanım Durumunu içermektedirler.



Sistemdeki Nesnelerin Ön Belirlenmesi (Identifying Initial Analysis Objects)

- Müşteriler, kullanıcılar ve sistem geliştiricileri sistemin spesifikasyonunu belirlemek için birlikte çalışırken karşılaştıkları en büyük zorluklardan biri kullandıkları farklı terminolojilerdir. Aynı ifadenin farklı durumlarda farklı manalarda kullanılması, ya da aynı şeyi ifade etmek için farklı terimlerin kullanılması yanlış anlamalara yol açabilmektedir.
- Bunu önlemek için Kullanım Durumları belirlendikten sonra, bu Kullanım Durumları'nda rol alan nesneler belirlenir. Bu nesneler, belirsizlikleri ortadan kaldıracak şekilde adlandırılıp tanımlandıktan sonra lügatçede (glossary) derlenir.



Sistemdeki Nesnelerin Ön Belirlenmesi (Identifying Initial Analysis Objects)

- Aşağıda Kullanım Durumları'nda rol alan nesneleri belirlemek için önerilmiş bazı bulușsal yöntemler (heuristic methods) gösterilmiştir:

Bulușsal Yöntemler

- Kullanıcı ve sistem geliştiricilerinin Kullanım Durumunu anlamak için açıklığa kavuşturmak zorunda oldukları kavramlar.
- Kullanım Durumlarında tekrar eden isimler (Ör. Vaka)
- Sistemin takibini yapmak zorunda olduğu gerçek hayat nesneleri (Ör. Bölge Sorumlusu, Görev Dağıtıcısı)
- Kullanım Durumları (Ör. AcilDurumBildirimi)
- Veri kaynakları (Ör. Yazıcı)
- Arayüz birimleri (İstasyon)
- Her zaman uygulama alanının kavramları kullanılmalı.



Tekrar

İşlevsel Olmayan İsterlerin Belirlenmesi (Identifying NonFunctional Requirements)

- İşlevsel olmayan isterler, kullanıcı tarafından fark edilebilen ama doğrudan sistemin işlevleri ile ilgili olmayan isterlerdir.
- İşlevsel olmayan isterler kullanıcı arayüzü görüntüsü, tepki süresi (response time), sistemin güvenliği gibi birçok konuya içine alır. İşlevsel olmayan isterler, işlevsel isterlerle eş zamanlı olarak belirlenir, çünkü işlevsel olmayan isterlerin de sistemin geliştirilmesinde ve maliyetinde işlevsel isterler kadar etkilidir.
- Bütün bu işlevsel olmayan isterler belirlendikten sonra önem sırasına koyulur ve en önemli olanlarına öncelik verilir.

- 1.Kullanıcı Arayüzü ve İnsan Faktörü
- 2.Dokümantasyon
- 3.Donanım Faktörleri
- 4.Performans Özellikleri
- 5.Hata Kontrolü ve İstisnai Durumlar
- 6.Kalite Konuları
- 7.Sistem Değişikliği
- 8.Fiziksel Çevre Koşulları
- 9.Güvenlik Konuları
- 10.Kaynak Konuları



İsterlerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

- Bir önceki bölümde, kullanım durumları açısından bir sistemin modellenmesiyle ilgili teknik konuları açıkladık.
- Ancak, kullanım durumları modellemesini kendi başlarına kullan, gerekliliklerin karşılanması gerektiğini gerektirmez. Uzman model kullanıldıktan sonra bile, geliştiriciler hala kullanıcıların gereksinimlerini ortaya çıkarmalı ve müşteri ile bir anlaşmaya varmalıdır. Bu bölümde, kullanıcılarından bilgi edinme ve bir müşteri ile anlaşma görüşme yöntemleri açıklanmaktadır.
- Bu başlık altında sistem geliştiricilerinin kullanıcılarından gerekli bilgiyi edinmeleri ve müşteri ile anlaşmaya varabilmelerinin yöntemleri üzerinde duracağız. Dört konu üzerinde yoğunlaşacağız:
 - Müşterilerle Şartların Müzakere Edilmesi: Ortak Uygulama Tasarımı (JAD)
 - İsterlerlerin doğrulanması (Kullanılabilirlik Testleri- Usability Testing)
 - Kullanıcılardan gerekli bilgiyi edinmek (KAT-Knowladge Analysis of Task)
 - Belirlenen isterlerin Dokümentasyonu



İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

- KAT (Knowledge Analysis of Tasks) metodunda sistem geliştiricileri, sistemi modellemek için gereken bilgileri sistemin uygulama alanını gözlemleyerek edinirler.
- Bu gözlemlerinde protokol analizleri, standart prosedürler, kitaplar, kullanıcı ve müşterilerle yaptıkları mülakatlar, çeşitli dokümanlar gibi kaynaklardan faydalanaırlar.
- Sonuçta sistemi nesneye dayalı programlamaya uygun olarak nesneler, eylemler, prosedürler, hedefler (goals) ve alt hedefler (sub goals) olarak modellerler. KAT beş adımda özetlenebilir:



İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

1. Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

Nesne ve Eylemlerin Belirlenmesi	Nesne ve Eylemlerin Belirlenmesi (Identifying Objects and Actions)
Prosedürlerin Belirlenmesi	Nesne ve eylemler nesneye dayalı (object-oriented) analizde olduğu gibi protokol analizleri, standard prosedürler, kitaplar, kullanıcı ve müşterilerle yapılan mülakatlar, çeşitli dokümanlar gibi kaynaklar incelenerek belirlenir.
Hedef ve Alt-Hedeflerin Belirlenmesi	
Önem ve Sıradanlığın Belirlenmesi	
Görevin Modelinin Oluşturulması	

İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

1. Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

Nesne ve
Eylemlerin
Belirlenmesi

Prosedürlerin
Belirlenmesi

Hedef ve Alt-
Hedeflerin
Belirlenmesi

Önem ve
Sıradanlığın
Belirlenmesi

Görevin
Modelinin
Oluşturulması

Prosedürlerin Belirlenmesi (Identifying Procedures)

Prosedür, bir eylemler kümesidir. Bir prosedürün başlaması için bir önkoşula (precondition) ve artkoşula (postcondition) ihtiyaç vardır. Prosedürdeki eylemler, kısmen sıralı olabilir. Prosedürler, senaryolar yazarak, kullanıcının yaptığı işlemler gözlemlenerek, kullanıcılarla mülakat yaparak belirlenebilir.

İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

1. Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

Nesne ve Eylemlerin Belirlenmesi
Prosedürlerin Belirlenmesi
Hedef ve Alt-Hedeflerin Belirlenmesi
Önem ve Sıradanlığın Belirlenmesi
Görevin Modelinin Oluşturulması

Hedef ve Alt-Hedeflerin Belirlenmesi (Identifying Goals and Sub-Goals)

Bir görevin başarıyla gerçekleştirilmesi için ulaşılması gereken duruma **hedef** (goal) denir. Hedefler, kullanıcılarla mülakat yaparak belirlenebilir. Alt-hedefler, goller ayırtılı olarak belirlenir.

İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

1. Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

Nesne ve
Eylemlerin
Belirlenmesi

Prosedürlerin
Belirlenmesi

Hedef ve Alt-
Hedeflerin
Belirlenmesi

Önem ve
Sıradanlığın
Belirlenmesi

Görevin
Modelinin
Oluşturulması

Önem ve Sıradanlığın Belirlenmesi (Identifying Typicality and Importance)

Belirlenen her nesneye, ne kadar sık rastlandığına
ve hedefe erişmek için ne kadar gerekli olduğuna
bağlı olarak bir değer biçilir.

İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

1. Kullanıcılardan Gerekli Bilgiyi Edinmek (KAT-Knowledge Analysis of Tasks)

Nesne ve
Eylemlerin
Belirlenmesi

Prosedürlerin
Belirlenmesi

Hedef ve Alt-
Hedeflerin
Belirlenmesi

Önem ve
Sıradanlığın
Belirlenmesi

Görevin
Modelinin
Oluşturulması

Görevin Modelinin Oluşturulması (Constructing a Model of the Task)

Yukarıdaki adımlarda elde edilen bilgilerden
sistemin modeli oluşturulur.

İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

2. Müşteri ile Sistem Spesifikasyonu Üzerinde Anlaşmaya Varmak (JAD-Joint Application Design)

- Ortak Uygulama Tasarımı (JAD), 1970'lerin sonunda IBM'de geliştirilen bir gereksinim yöntemidir. Etkinliği, ihtiyaçların karşılanması çalışmasının tüm paydaşlarının katıldığı tek bir atölye çalışmasında yapılmasıdır.
- Kullanıcılar, müşteriler, geliştiriciler ve eğitimli bir oturum lideri, bakış açılarını sunmak, diğer bakış açılarını dinlemek, müzakere etmek ve karşılıklı olarak kabul edilebilir bir çözüme ulaşmak için bir odada bir arada oturlar.



İsterlerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

2. Müşteri ile Sistem Spesifikasyonu Üzerinde Anlaşmaya Varmak (JAD-Joint Application Design)

- Son JAD belgesi olan çalıştayın sonucu, veri elemanlarının, iş akışlarının ve arayüz ekranlarının tanımlarını içeren eksiksiz bir şartname belgesidir.
- Nihai belge, paydaşlar tarafından ortaklaşa geliştirildiğinden (yani, sadece projenin başarısı ile ilgilenmeyen, aynı zamanda önemli kararlar alabilen katılımcılar), nihai JAD belgesi, kullanıcılar, müşteriler ve geliştiriciler, ve böylece geliştirme sürecinde daha sonra gereksinimlerini en aza indirir. JAD beş faaliyetten oluşmaktadır



İsteklerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

2. Müşteri ile Sistem Spesifikasyonu Üzerinde Anlaşmaya Varmak (JAD-Joint Application Design)



İsterlerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

3. İsterlerin Doğrulanması

- Belirlenen Kullanım Durumları'nın, yani sistem modelinin doğrulanması için kullanıcılarla kullanılabilirlik testleri düzenlenir. Kullanıcılardan sistemin bir kısmını veya tamamını kullanmaları ve görüşlerini bildirmeleri istenir.

Senaryo testi	Prototip testi	Ürün testi
Senaryo testi (Scenario test) Bu testte, bir veya birkaç kullanıcıya oluşturulan Senaryolar kağıt ortamında ekran çıktıları veya küçük prototipler şeklinde sunulur. Kullanıcıların bu Senaryoları ne kadar çabuk ve kolay anlayabildikleri değerlendirilir.		
Senaryo testi	Prototip testi	Ürün testi
Prototip testi (Prototype test) Bu testte, kullanıcılar sistemin prototipi sunulur ve kullanıcılarından bu prototipi kullanıp görüşlerini bildirmeleri istenir. Prototip testlerinin avantajı sistemi gerçekçi olarak ele almalarıdır. Dezavantaj ise prototiplerin oluşturulmasının pahalı olmasıdır.		
Senaryo testi	Prototip testi	Ürün testi
Ürün testi (Product test) Ürün testleri de prototip testlerine benzerler, fakat sistemin prototipini değil de işlevsel bir versyonunu kullanırlar. Sistem, test sonuçlarındaki görüşlerin değerlendirilebilmesi için kolay değiştirilebilir bir nitelikte olmalıdır.		



İsterlerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

4. Belirlenen İsterlerin Dokümantasyonu

- İsterlerin belirlenmesi ve analizi sonucunda **İsterler Analiz Dokümanı** (RAD-Requirements Analysis Document) yazılır.
- Bu doküman sistemin işlevsel ve işlevsel olmayan bütün isterlerini içerir ve müşteri ile sistem geliştiricileri arasında bir kontrat özelliği taşır. RAD, müşteriye, kullanıcılarla, proje yöneticilerine, sistem tasarımcılarına ve analistlere hitap eder. Dokümanın Kullanım Durumları ve işlevsel olmayan isterleri içeren ilk kısmı isterlerin belirlenmesi aşamasında yazılır.
- Nesne modelleri ise isterlerin analizi aşamasında yazılır. RAD, Kullanım Durumları sabitlenince, yani sistemdeki olası değişiklikler minimum düzeye inince yazılmalıdır. Fakat, sistem geliştirilirken oluşan bütün değişiklikleri yansıtacak şekilde güncellenmelidir.



İsterlerin Belirlenmesi Aşamasının Yönetimi (Managing Requirements Elicitation)

4. Belirlenen İsterlerin Dokümantasyonu

Requirements Analysis Document

1. Introduction
 - 1.1 Purpose of the system
 - 1.2 Scope of the system
 - 1.3 Objectives and success criteria of the project
 - 1.4 Definitions, acronyms, and abbreviations
 - 1.5 References
 - 1.6 Overview
2. Current system
3. Proposed system
 - 3.1 Overview
 - 3.2 Functional requirements
 - 3.3 Nonfunctional requirements
 - 3.3.1 Usability
 - 3.3.2 Reliability
 - 3.3.3 Performance
 - 3.3.4 Supportability
 - 3.3.5 Implementation
 - 3.3.6 Interface
 - 3.3.7 Packaging
 - 3.3.8 Legal
 - 3.4 System models
 - 3.4.1 Scenarios
 - 3.4.2 Use case model
 - 3.4.3 Object model
 - 3.4.4 Dynamic model
 - 3.4.5 User interface—navigational paths and screen mock-ups
4. Glossary