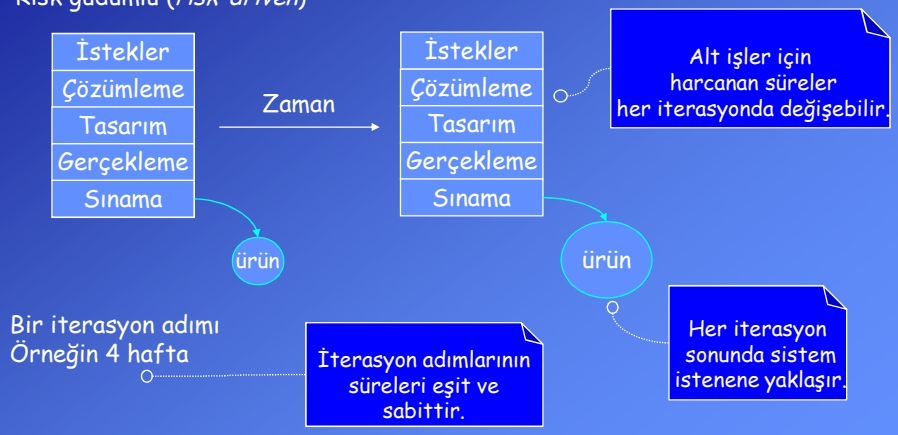
Tümleştirilmiş Yazılım Geliştirme Süreci (*The Unified Process* – UP)

Deneyimler sonucu kabul gören en iyi özellikler bir araya getirilmiştir.
- Yinelemeli (*iterative*)
- Arttırmalı ve evrimsel (*incremental, evolutionary*)
- Risk güdümlü (*risk-driven*)



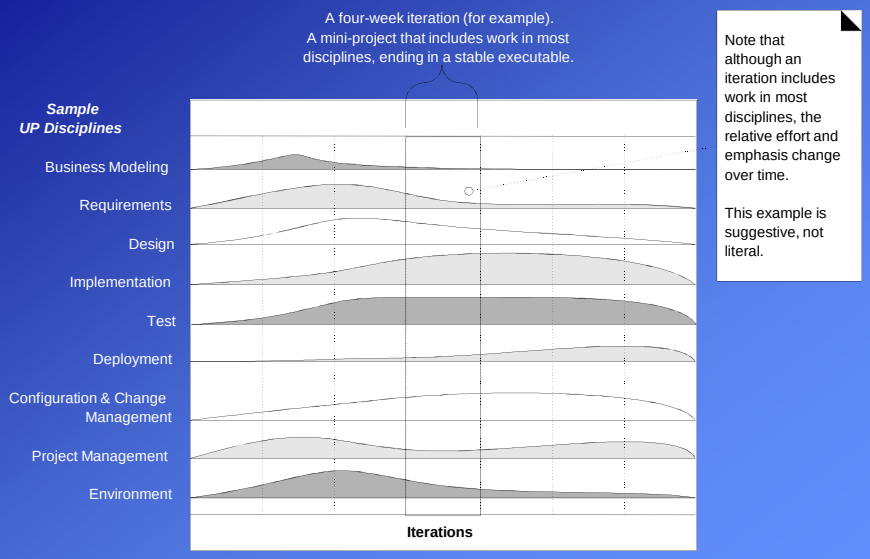
Yinelemeli Sürecin Yararları

- Değişen isteklere uyum
- Erken geri besleme
- Büyük sistemlerde çözümleme kolaylığı
- Her iterasyonda deneyim kazanılması
- Risklerin erken giderilmesi (eğer mümkünse)
- Erken ürün elde etme, takımda moral yükselmesi

Öneriler:

- 2 - 6 haftalık sabit süreli iterasyonlar uygulanmalı
- Yüksek risk taşıyan kısımlar ilk iterasyonlarda gerçeklenmeli
- Temel oluşturan yapılar (çekirdek) önce gerçeklenmeli
- Sürekli kullanıcılardan geri besleme alınmalı, isteklere uyulmaya dikkat edilmeli
- Her iterasyondan sonra ürün tam olarak sınanmalı
- Kullanım senaryoları yöntemi (*use case*) uygulanmalı
- Görsel modelleme (UML) kullanılmalı
- Bir iterasyonda elde edilen deneyim diğer iterasyonda kullanılmalı

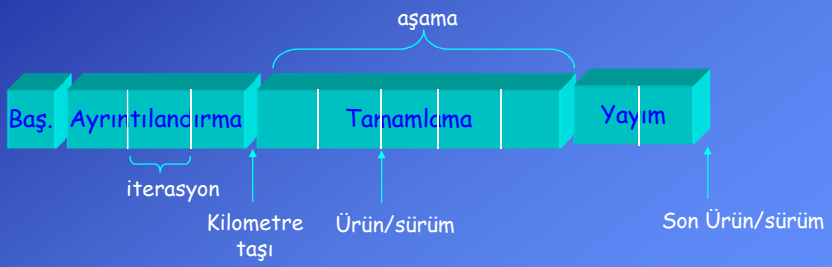
İşlerin İterasyonlara Dağılımı



UP Aşamaları

Tümleştirilmiş Süreçte (UP) yazılım geliştirme aşamaları:

- Başlangıç (*Inception*): Kabaca vizyon, fizibilite, tamam/devam?
- Ayrıntılandırma (*Elaboration*): Daha gerçekçi çözümleme, çekirdek yapının ve yüksek riskli kısımların yinelemeli olarak oluşturulması.
- Tamamlama (*Construction*): Daha az riskli ve düşük öncelikli kısımların yinelemeli olarak gerçeklenmesi.
- Yayım (*Transition*): Beta testleri, piyasaya sürme çalışmaları.

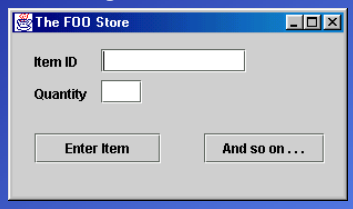


Derste Kullanılacak Örnek Sistem

Dersteki örnekler çoğunlukla "NextGen" adı verilen bir POS (*point of sale*) sistemi üzerinde verilecektir. Örnek, dersin ana kitabından alınmıştır: Craig Larman, Applying UML and Patterns , An Introduction to OOA/D and Iterative Development, 3/e, 2005.

Yazılımın üç katmandan oluştuğu düşünülmüştür.

Arayüz (*Interface*)



Üzerinde fazla durulmayacak, diğer katmanlarla bağlantının nasıl sağlanacağı incelenecek

Uygulama Lojiği (*application logic and domain object layer*)

Üzerinde çalışılacak olan temel katman. Nesnelerin tasarımı yapılacak.

Teknik Hizmetler (*technical services layer*)

Üzerinde çalışılacak olan ikinci öncelikli katman. Nesnelerin tasarımı yapılacak.

İsteklerin Çözümlenmesi (*Requirement Analisys*)

İstekler bir sistemin sahip olması gereken yetenekler ve sağlaması gereken koşulların tarifidir.

İstekler **FURPS+** modeli ile kategorize edilebilir.

- İşlevsellik (*Functionality*): İşlevleri, yetenekleri, güvenlik
- Kullanılabilirlik (*Usability*): İnsan ile etkileşimi, yardım, dokümantasyon
- Güvenirlilik (*Reliability*): Hataların sıklığı, düzeltilebilmesi, öngörülebilmesi
- Performans (*Performance*): Hız, verimlilik, doğruluk
- Desteklenebilme (*Supportabilty*): Güncelleme, bakım, seçeneklerin ayarlanabilmesi, uluslararası uyum
- +: Diğer ikincil faktörler:
  - Gerçekleme: Kullanılan dil, araçlar, kaynaklar
  - Arayüz: Diğer birimler ile etkileşimi
  - İşletme: İşletme (kullanım) ile ilgili beklentiler
  - Paketleme
  - Lisans: Hukuki işlemler

İstekler işlevsel olanlar ve diğerleri olarak iki gruba da ayrılabilir.

Bu kategoriler isteklerin unutulmasını önlemek için kullanılır.

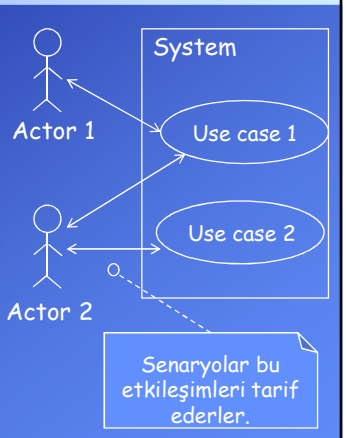
Kullanım Senaryoları (*Use-Case Model*)

İsteklerin anlaşılmasını ve ifade edilmesini sağlayan bir yöntemdir.

Özellikle işlevsel isteklerin ifade edilmesinde kullanılır.

Senaryolar sadece bir doküman değildir.

Senaryolar olmadan sistemin ne yapması gerektiği ne olarak belirlenemez.



Ivar Jacobson, İsveçli müh. (Ericsson), sonra Rational'de, şimdi kendi firmasında.

http://www.ivarjacobson.com

Tanım:

- "A use case specifies a sequence of actions, including variants, that a system performs and that yields an observable result of value to a particular actor." (Three amigos: Jacobson, Booch, Rumbaugh 1999)
- "A use case is a collection of possible sequences of interactions between the system under discussion and its external actors, related to a particular goal." (Cockburn 2000)

Senaryo : Anlamlı bir sonuca (amaca) ulaşmak için aktör ile sistem arasında gerçekleşen olayların belli bir zinciridir. Bir sistemin çalışması sırasında birden fazla senaryo gerçekleşebilir.

Olası tüm senaryolar kullanım senaryolarını (*use case*) oluştururlar.

Örnek:

Bir otomatik para çekme makinesinde (ATM) müşteri ile sistem arasında gerçekleşebilecek olan olayların oluşturduğu senaryolar şunlar olabilir.

1. Müşteri kartını makineye takar.
2. Sistem şifreyi sorar.
3. Müşteri şifreyi girer.
4. Sistem şifreyi onaylar.
5. Müşteri para çekme işlemini seçer.
6. Müşteri çekeceği para miktarını seçer.
7. Sistem parayı, makbuzu ve kartı verir.

Yukarıdaki akış bu sistemdeki olası senaryolardan sadece biridir.

Aynı sistemdeki başka bir senaryo da müşterinin bakiyesinin yeterli olmaması durumuyla ilgilidir.

Aktör: Sistemin kullanıcılarını tanımlamak için kullanılan mekanizmadır.

- Aktör tasarlanmakta olan sitemin kullanıcısı ya da o sistemden etkilenen diğer birimlerdir; insan, başka bir sistem, bir cihaz olabilir.
- Aktörler tasarlanacak olan sistemin dışında kalan birimlerdir.
- Aktör sistemden hizmet isteğinde bulunabilir, sisteme hizmet verebilir.

Farklı gruplara ayrılırlar:

- Birincil Aktör (*Primary Actor*): Sistemden asıl faydayı sağlayan, işlemleri başlatan kullanıcı.
- Destek Aktörü: Sisteme bilgi (destek) sağlayan aktör. Genellikle bir bilgisayar sistemidir.
- Diğer Aktörler: Bu aktörler sistemi doğrudan kullanmazlar ve sisteme bilgi desteği vermezler ancak o senaryoda gerçekleşen olaylarla ilgilenirler ve bu olaylardan etkilenirler.

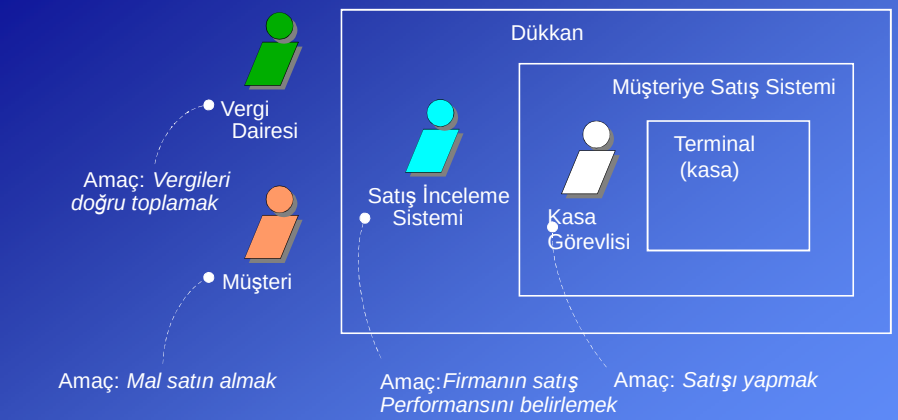
Aktörlere ilişkin örnekler derslerin ilerleyen bölümlerinde verilecektir.

Birincil Aktör ve Sistemin Sınırları:

Üzerinde çalıştığımız sistemi hangi düzeyde incelediğimize ve sınırlarını ne şekilde çizdiğimize bağlı olarak birincil aktörler değişiklik gösterir.

Kullanım senaryolarını yazarken sistemin sınırlarını doğru olarak belirlemek, nelerin dışarıda nelerin içeride olacağına doğru karar vermek gerekir.

Birincil Aktör ve Sistemin Sınırları



Şekilde görüldüğü gibi o anda tasarlamakta olduğumuz sistem sadece terminal programı ise bu durumda sistemin birincil aktörü (kullanıcısı) kasa görevlisidir.

Ancak dükkan sistemini bir bütün olarak inceliyorsak kasa görevlisi bu sistemin içinde bir parçadır ve aktör değildir. Bu durumda birincil aktör müşteridir. Vergi dairesi ise bu sistemden etkilenen diğer aktördür.

Kullanım Senaryolarının Yazılması

Kullanım senaryolarının ifade edilmesi:

- İhtiyaçların ve istenen özelliklerin listelenmesi şeklinde **DEĞİL**.
- Sistem **kara kutu** olarak ele alınır. Sistemin iç yapısı görülmez, sistemin dışarıya (aktörlere) karşı sorumlulukları ifade edilir.
- Aktörler ile sistem arasındaki etkileşim **etken** cümleler ile ifade edilir.
- "Ne yapar?" sorusu cevaplanır, "Nasıl yapar?" **değil**. Sistemin sorumluluklarını nasıl yerine getireceği daha sonra gelinecek olan tasarım aşamasında ele alınacak problemdir. Kullanım senaryolarını yazdığımız şimdiki aşamada ise sadece istekler anlaşılmaya çalışılıyor.
- Sistemin bitmiş hali hayal edilerek bu sistem çalıştığında oluşabilecek senaryolar yazılır.

Kullanım senaryolarında yer alan bölümler:

Her kullanım senaryoları grubunun (*use case*) bir adı ve numarası vardır. İsimden sonra aşağıdaki bölümler gelir.

a) Önsöz (*Preface*) Bölümü

Aşağıdaki alt bölümlerden oluşur:

- Birincil Aktör (*Primary Actor*): Sistemden asıl faydayı sağlayan, işlemleri başlatan kullanıcı.

- İlgililer ve Beklentileri (*Stake holders and interests*): Sistemin çalışmasından etkilenen ve bu sistemden beklentileri olan unsurlar (diğer aktörler).

Birincil aktör, destek aktörü ve diğer aktörlerin belirlenmesi sistemin sınırlarını çizer.

Kullanım senaryoları ilgililerin (aktörlerin) tüm beklentilerini karşılayan tüm olayları ve sadece onları içerir.

Tüm ilgililerin ve beklentilerin ilk başta belirlenmesi önemlidir. Aksi durumda senaryolarda bazı durumlar unutulabilir ve bu eksiklik ancak ileriki aşamalarda anlaşılabilir.

- Ön koşullar (*Preconditions*): Belli bir senaryo grubunu (*use case*) oluşturan olayların başlaması için sağlanması gereken koşullar.

Bu koşullar senaryo içinde test edilmez, doğru oldukları varsayılır.

- Son koşullar (*Postconditions, Success Guarantees*): Senaryolar tamamlandığında sistemin ulaşacağı durumlardır.

Son koşullar ilgililerin beklentilerine (amaçlarına) denk düşer.

b) Ana Başarılı Senaryo (Temel Akış) Bölümü (*Main Success Scenario or Basic Flow*

Sistemin en doğal çalışma şekli adım adım yazılır. Her adım numaralanır.
Koşullar ve dallanmalar içermez. **Etken cümleler** kullanılır; "kim ne yapar" açıktır.
Adımlar üç farklı gruba ayrılır:
1. Kullanıcılar ile sistem arasında etkileşim, tetikleme.
2. Onaylama (çoğunlukla sistem tarafından)
3. Sistemde durum değişikliği, bir bilginin kayıt edilmesi.

Örnek:
1. Müşteri şifresini girer.
2. Sistem ekrana müşterinin adını çıkartır.
3. ....

Belirsiz ve edilgen cümleler kullanılmaz. Örnek: "Toplam belirlenir." Bu uygun bir senaryo cümlesi değildir. Kim belirleyecek? Sistem mi? Aktörlerden biri mi?

c) Uzantılar (Alternatif Akışlar) Bölümü (*Extensions or Alternate Flows*)

Ana senaryonun dışında kalan başarılı/başarısız sonuçlara götüren tüm senaryolar sıralanır.
Ana senaryodan (temel akış) dallanmalar şeklinde yazılırlar. Ana senaryoda hangi adımdan buraya gelinecekse o adımın numarası kullanılır.
Alternatif akışa (dallanma) neden olan koşullar aktörler ya da sistem tarafından fark edilecek şekilde yazılmalı.
Alternatif senaryolar ile aktörlerin tüm amaçları sağlanmış olmalı.

Örnek: Ana senaryoda "2. Müşteri şifresini girer" satırı varsa, temel akışta şifrenin doğru olduğu durum ele alınır. Şifrenin yanlış girilmesi durumu ise aşağıda gösterildiği gibi uzantılarda incelenir.

Uzantılar:

2a. Müşteri şifresini yanlış girmiştir.
1. Sistem hata mesajı verir ve şifreyi yeniden ister.

d) Sıra Dışı Durumlar Bölümü (*Exceptions*)

Sistemde hatalar oluştuğunda yapılacaklar sıralanır.
Bazı tasarımcılar bu bölümdeki olayları da uzantılar bölümünde ele alırlar.

e) Özel İstekler Bölümü (*Special Requirements*)

İşlevler ile ilgili olmayan istekler bu bölümde belirtilir.

Bu istekler genellikle hız, güvenirlilik, rahat kullanım gibi kalite kriterlerine yöneliktir.

f) Teknolojik Beklentiler Bölümü

Kullanıcıların ön gördükleri donanım özellikleri burada sıralanır.

Örneğin giriş/çıkış işlemlerinin hangi cihazlar ile yapılması istendiği bu bölüme yazılır.

Örnek:
Metin (text) tipindeki bir kullanım senaryoları grubuna örnek olarak bir marketteki satış noktası (POS) uygulaması verilmiştir. Bir sistemde bir çok senaryo grubu (*use case*) bulunabilir.
Örneğin market sisteminde de satış işlemleri bir senaryolar grubu, ürün iadesi de başka bir senaryolar grubu olabilir.

Bu örnekte satış işlemleri (*Process Sale*) senaryo grubu gösterilmiştir.

Senaryo Grubu (*Use Case*) SG1: Satış İşlemleri:

Konu: NextGen POS Market Sistemi
Birincil Aktör: Kasa Görevlisi
İlgililer (Aktörler) ve Beklentileri (*Stakeholders and Interests*):
• Kasa Görevlisi: Bilgilerin doğru ve hızlı girilmesi, toplamın doğru hesaplanması, para üstünün doğru hesaplanması
• Satış Elemanı: Komisyonun doğru hesaplanması ve kayıt edilmesi
• Müdür: Yetkili işlemleri (kasa görevlisinin yapamadığı) kolaylıkla yapabilmek
• Vergi Dairesi: Vergilerin doğru hesaplanabilmesi ve toplanabilmesi
• Kredi Kartı Asıllama Merkezi: Ödeme bilgilerinin doğru formatta gelmesi ve asıllama bilgilerinin kayıt edilmesi

Ön Koşullar (*Preconditions*): Kasa görevlisi sisteme giriş yapmıştır.
Son Koşullar (*Postconditions*): Satış bilgileri kayıt edilmiştir. Vergi doğru olarak hesaplanmıştır. Muhasebe ve envanter kayıtları güncellenmiştir. Komisyon kayıt edilmiştir. Fatura oluşturulmuştur. Kredi kartı onayı kayıt edilmiştir.

Ana Başarılı Senaryo (Doğal Akış) (*Main Sucess Scenario or Basic Flow*) :

1. Müşteri ödeme noktasına almak istediği ürün ve hizmetler ile gelir.
2. Kasa görevlisi yeni bir satış başlatır.
3. Kasa görevlisi ürün kodunu sisteme girer.
4. Sistem satış kalemini (maddesini) kayıt eder ve ürünün tanıtıcı bilgisini, fiyatını ve o anda kadar oluşan toplamı gösterir.

Kasa görevlisi 3ncü ve 4ncü maddeleri ürün kalmayıncaya kadar tekrar eder.

5. Sistem toplamı vergilerle birlikte gösterir.
6. Kasa görevlisi müşteriye toplamı söyler ve ödeme yapmasını ister.
7. Müşteri ödeme yapar ve sistem ödeme bilgilerini alır.
8. Sistem tamamlanan satış bilgilerini kayıt eder; satış ve ödeme ile ilgili bilgileri muhasebe ve envanter sistemlerine (bunlar dış sistemlerdir) gönderir.
9. Sistem faturayı oluşturur.
10. Müşteri ürün ve hizmetler ile ayrılır.

Uzantılar (Alternatif Akışlar) (*Extensions or Alternate Flows*):

*a. Herhangi bir anda müdür yetkili bir işlem yapmak ister ve şifresini girer:
   1. Sistem müdür-yetkisi konumuna geçer.
   2. Müdür yetkili bir işlem gerçekleştirir. Örneğin satışı iptal eder, bir ürünün fiyatını indirir vs.
   3. Müdür sistemden çıkar.
   4. Sistem normal konuma (kasa görevlisi yetkisi) geçer.

Uzantılar (Alternatif Akışlar) (*Extensions or Alternate Flows*) Devamı:

*b. Herhangi bir anda sistemde bir hata oluşur:
   Bu durumlarda bilgilerin kayıt edilmesi ve sistemin kaldığı yerden devam edebilmesi istenir.
   1. Kasa görevlisi sistemi yeniden başlatır, sisteme giriş yapar ve sistemin önceki durumdan devam etmesini ister.
   2. Sistem önceki durumu oluşturur.
      2a. Sistem önceki durumu oluştururken anormallik sezer.
         1. Sistem hata uyarısı verir, hatayı kayıt eder ve temiz (başlangıç) duruma geçer.
         2. Kasa görevlisi yeni bir satış başlatır.

3a. Geçersiz bir ürün kodu (Sistemde bulunamadı):
   1. Sistem hata uyarısı verir, ürünü reddeder.
   2. Kasa görevlisi hataya tepki verir:
      2a. Ürünün üstünde okunabilir bir kod vardır:
         1. Kasa görevlisi kodu sisteme elle (*manual*) girer.
         2. Sistem ürünün tanıtıcı bilgisini ve fiyatını gösterir.
      2b. Ürünün üstünde kod yoktur, ama fiyatı yazılıdır:
         1. Kasa görevlisi müdürden yetkili bir işlem yapmasını ister.
         2. Müdür şifresini girer.
         3. Kasa görevlisi fiyatı elle girer.

3b. Aynı üründen bir taneden fazla alınmıştır (5 şişe içecek):
1. Kasa görevlisi ürün kodunu ve adetini sisteme girer.

3-6a. Müşteri kasa görevlisine bir ürünü almaktan vazgeçtiğini söyler:
1. Kasa görevlisi satıştan çıkarılacak ürünün kodunu sisteme girer.
2. Sistem ürünü satıştan çıkarır ve geçerli toplamı gösterir.

3-6b. Müşteri alışverişten vazgeçtiğini söyler:
1. Kasa görevlisi satışı iptal eder.

5a. Müşteri indirim hakkı olduğunu söyler (müşteri kartına sahiptir):
1. Kasa görevlisi müşteri kodunu sisteme girer.
2. Sistem indirimi uygular ve yeni toplamı gösterir.

7a. Nakit ödeme:
1. Kasa görevlisi ödenen nakit miktarı sisteme girer.
2. Sistem para üstünü gösterir ve para çekmecesini açar.
3. Kasa görevlisi müşteriden ödemeyi alır ve para üstünü verir.
4. Sistem nakit ödemeyi kayıt eder.

7b. Kredi kartı ile ödeme:
1. ....

7c. Çek ile ödeme:
1. ....

Özel İstekler (*Special Requirements*):
- Düz kare monitör. Yazılar 1 metre uzaklıktan okunabilmeli.
- Kredi kartı sorgulamasının cevabı en geç 30 saniyede gelmeli.
- ....

Teknolojik Beklentiler (*Technology Variations List*):

*a. Müdür kendisini sisteme bir kart okutarak ya da tuş takımından şifresini girerek tanıtır.

3a. Ürün kodları bir barkod okuyucu ile veya tuş takımından elle girilebilir.

7b. Kredi kartı bilgiler kart okuyucu ile veya tuş takımından elle girilebilir.

....

Açık noktalar (*Open Issues*):
- Vergi kanunlarındaki değişim sistemi nasıl etkiler?
- Kasa görevlisi mesaisi bittiğinde sistemden çıkarken para çekmecesini de almalı mı?
- Müşteri kart okuyucuları doğrudan kendisi kullanabilir mi, yoksa kasa görevlisi ile mi sisteme erişmeli?
- ....

UML Kullanım Diyagramları (*Use Case Diagram*)

Kullanım senaryoları sadece düz metin (*text*) olarak değil, istendiğinde metin yerine UML diyagramı olarak da ifade edilebilirler.

Kullanım diyagramlarında, kullanım senaryolarının aktörler ile ve kendi aralarındaki ilişkileri grafik olarak gösterilir.

Bir sistemin içinde bir çok senaryo grubu bulunabilmekte ve değişik aktörler değişik senaryo grupları ile ilişkili olabilmektedir.

Ayrıca senaryo gruplarının kendi aralarında da içerme (*include*) ve genişletme (*extend*) ilişkileri bulunabilmektedir.

1. İçerme (*includes, uses*): Birçok senaryo grubunda kullanılan başka bir senaryo grubudur. Örneğin otomasyon sistemini kullanmak için giriş yapılması gerekir.

Bir senaryonun içinden bir alt programa dallanıp geri dönmek gibidir.

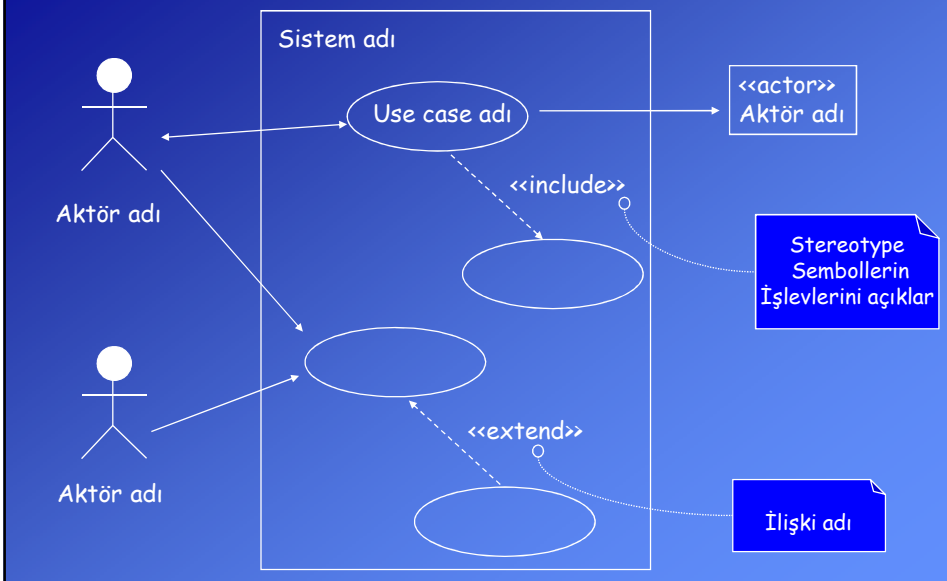
2. Genişletme (*extends*): Senaryo grupları doğal akışa göre hazırlanır. Çeşitli koşullar altında bu doğal akıştan sapmalar olabilir.

Genişletme ilişkisi ana senaryodan ayrılma noktasından sonra yapılanları belirtir.

UML diyagramlarında bir şeklin anlamını açıklayan özel sözcükler (*sterotype*) <<...>> simgeleri arasına yazılır.

Aktörler çizgi adam şeklinde gösterildiği gibi bir dikdörtgen ile de ifade edilebilir. Bu durumda dikdörtgenin anlamını belirtmek için "*streotype*" kullanılır

Use Case Diagram:



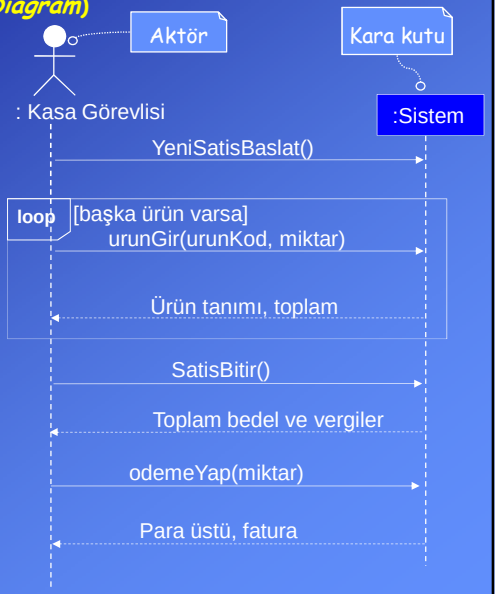
Etkileşim Diyagramı (*Interaction Diagram*)

Kullanım diyagramları sadece sistemde hangi senaryo gruplarının ve hangi aktörlerin yer aldığını gösterir.

Aktörler ile sistem arasında geçen olayları yani senaryoların adımlarını göstermek için etkileşim diyagramları (*interaction diagram*) çizilir.

Senaryoları ifade ederken aynı anda hem metin tipi senaryo yazımına hem de UML ile kullanım diyagramlarını ve etkileşim diyagramlarını çizmeye gerek yoktur.

Senaryoları belirtmek için metin tipi yazım ya da etkileşim diyagramı tercih edilir.



Örnek:

Örnek olarak bir öğrenci otomasyon sisteminin bir kısmına ait kullanım diyagramı gösterilmiştir.

Örnek istemin içinde dört adet senaryo grubu bulunmaktadır: Sisteme giriş, derse kayıt, geç kayıt ve sınıf listesi göster.

Bu senaryo gruplarının arasında çeşitli ilişkiler geçerlidir. Örneğin derse kayıt ve sınıf listesini göster senaryoları sisteme giriş senaryosunu içermektedir. Çünkü derse kayıt senaryoları yürütülürken sisteme giriş senaryosunun de yürütülmesi gereklidir.

Diğer taraftan geç kayıt senaryosu derse kayıt senaryosunu genişletmektedir. Normal işlemler derse kayıt senaryolarında belirtilmektedir. Eğer öğrenci belirtilen sürede kayıt olmamışsa geç kayıt senaryosuna geçilmektedir.

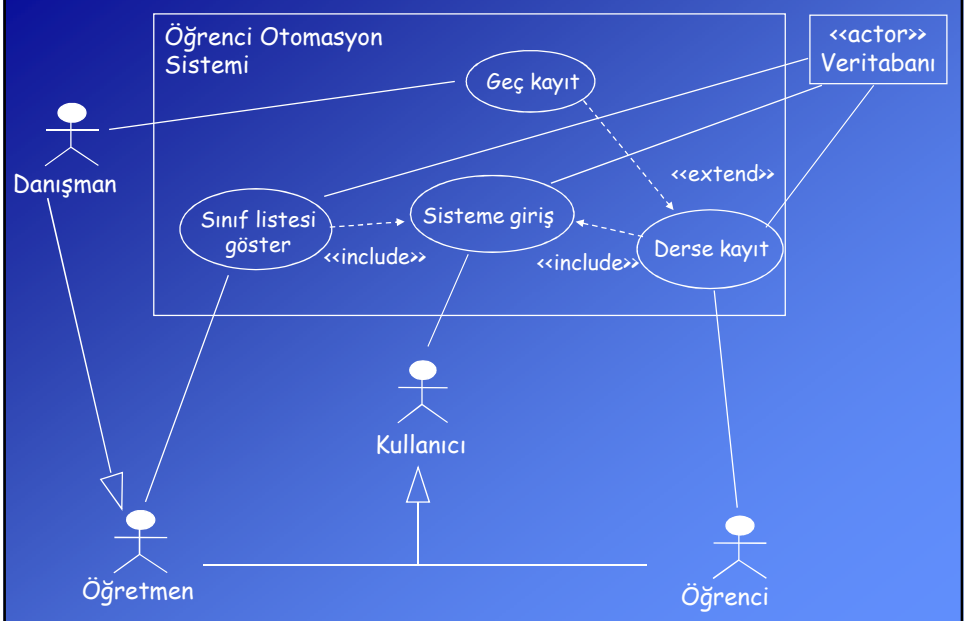
Ayrıca aktörlerin aralarında da nesneye dayalı programlamadan anımsayacağımız genelleşme/özelleşme (*generalization/specialization*) ilişkisi bulunabilmektedir.

Örnekte kullanıcı adını verdiğimiz bir aktör vardır. Öğrenci ve öğretmen bu kullanıcının özel halleridir. Danışman ise öğretmen aktörünün özel bir halidir.

Kullanım diyagramı incelendiğinde tüm kullanıcıların (öğretmen ya da öğrenci) sisteme giriş senaryolarında aynı şekilde rol oynadıkları görülür.

Dese kayıtta ise öğrenci, sınıf listesini göster senaryolarında ise öğretmen aktörleri rol oynamaktadır.

Geç kayıt senaryolarında öğretmen aktörünün özel bir hali olan danışman aktörü yer almaktadır.



Kullanım Senaryolarının Yazılması

Kullanım senaryoları, yazılımın müşterisi ile görüşülerek yazılır.

Müşteri bazen yazılımı yapan firmanın kendisi de olabilir.

Tasarıma geçmeden önce bütün isteklerin belirlenmesi gerekli (ve olası) değildir.

İstatistiklere göre proje süresinde isteklerin %25'i değişebilmektedir.

Müşteri ile görüşmelerde sistemin nasıl çalışacağı (iş akışları) açıkça ortaya konmalıdır.

Yazılımı doğrudan kullanacak olan kişilerle de görüşmeler yapılmalıdır.

Bu görüşmelerde aşağıdaki sorular sorularak senaryoların yazılmasında gerekli olan bilgilere ulaşılabilir.

Aktörlerin belirlenmesi için sorulabilecek sorular:

- Sistemin temel işlevlerini kim kullanacak?
- Günlük işlerini yapmak üzere kim sistemin desteğine gerek duyar?
- Sistemin bakımını ve işletmesini kim yapacak?
- Sistem hangi cihazları kullanacak?
- Hangi diğer sistemler ile etkileşimde bulunacak?
- Bu sistemin sonuçları kimi ilgilendirir?

Aktörlerden yararlanarak sistem davranışının belirlenmesi için sorulabilecek sorular:

- Aktörlerin temel işleri nedir?
- Aktör sistem bilgilerine erişmeli mi? Erişim tipi?
- Aktör dış durumlardaki değişiklikleri bildirecek mi?
- Durum değişiklikleri (hangileri?) aktöre bildirilecek mi?
- Aktör hangi işlevlere gerek duyar?

Diğer Sorular: Bazı davranışlar aktörlerden yola çıkarak belirlenemeyebilir. Bu durumda aşağıdaki soruları da sormakta yarar vardır:

- Sistemin gerek duyduğu girişler ve çıkışlar nelerdir?
- Sistem hangi dış olaylardan etkilenir?
- Şu andaki sistemin (eğer firmada aynı iş için kullanılan eski bir sistem varsa) eksikleri ve problemleri nelerdir?
- Periyodik olarak gerçekleştirilen işler var mı?

Senaryolar ve UP Yaklaşımı:

- UP'de, özellikle çevik yaklaşımda senaryoların tamamının projenin başında yazılması hedeflenmez (yinelemeli, evrimsel yaklaşım).
- Bir senaryonun gerçeklenmesi birden daha fazla yineleme adımı sürebilir. Daha küçük senaryolar tek bir adımda da gerçeklenebilir.

Kullanım Senaryoları Yönteminin Yararları

İsteklerin doğru ve eksiksiz olarak belirlenebilmesi yazılımın kalitesi açısından önemlidir. Kullanım senaryoları yöntemi bu noktada aşağıdaki yararları sağlar:

- Kolay anlaşılır. Müşteri (yazılımın kullanıcısı) ile yazılımı hazırlayacak grup arasında iletişimi kolaylaştırır.
- Sistemde gerekli olan unsurların belirlenmesini kolaylaştırır, unutulmalarını önler.
- Sınama (*verification*) olanağı sağlar. Gerçeklenen sistem senaryolar ile sınanabilir. Tamamlanmış olan yazılıma senaryolar uygulandığında eğer sistem her adımda senaryoda yazılmış olanları yerine getiriyorsa yazılımın sağlaması yapılmış olur.
- Kullanım senaryoları nesneye dayalı değildir. Bu yöntem gerekirse başka programlama yöntemleri için de kullanılabilir.
- Diğer taraftan kullanım senaryoları nesneye dayalı modelleme için uygun bir başlangıç noktası oluştururlar.

Bundan sonraki bölümlerde çözümleme ve tasarlama konuları anlatılırken kullanım senaryolarının bu yararı da gösterilecektir.

http://alistair.cockburn.us/usecases/usecases.html

http://www.pols.co.uk/use-case-zone/