TANITIM

Bu bölümde, gereksinimlerin bulunabileceği, keşfedilebileceği, ele geçirilebileceği veya zorlanabileceği birçok yolu keşfedeceğiz. Bu bağlamda, bu terimlerin tümü ortaya çıkarma ile eş anlamlıdır. Ancak "toplama" eşdeğer değildir. Gereksinimler, basitçe alınıp bir kile konacak düşen meyveler gibi değildir. Gereksinimlere ulaşmak genellikle o kadar kolay değildir, en azından hepsini değil. Daha incelikli ve karmaşık olanlardan birçoğunun, inatçı olmasa da titiz süreçlerle ayıklanması gerekir.

Gereksinim tespiti yapmak için seçebileceğiniz birçok teknik vardır ve muhtemelen farklı kullanıcı/paydaş sınıfları için birden fazla ve muhtemelen farklı olanları kullanmanız gerekecektir.

Tartışacağımız teknikler şunlardır:

- Beyin fırtınası
- Kart sıralama
- Çırak olarak tasarımcı
- Alan analizi
- Etnografik gözlem
- Hedefe dayalı yaklaşımlar
- Grup çalışması
- Röportajlar
- İçe bakış
- Ortak uygulama geliştirme (JAD)
- Merdiven
- Protokol analizi
- Prototipleme
- Kalite işlevi dağıtımı (QFD)
- Anketler
- Repertuar ızgaraları
- Senaryolar
- Görev analizi
- Kullanım durumları
- Kullanıcı hikâyeleri
- Bakış açıları
- Atölyeler

Bu liste, Zowghi ve Coulin (1998) tarafından önerilenden uyarlanmıştır.

GEREKSİNİMLERİN ORTAYA ÇIKARILMASI İÇİN HAZIRLANMA

Tüm müşterilerin ve paydaşların belirlenmesi, gereksinimlerin ortaya çıkarılmasına hazırlanmanın ilk adımıdır. Ancak paydaş grupları ve özellikle müşteriler homojen olmayabilir ve bu nedenle her bir alt gruba farklı davranmanız gerekir. Örneğin, evcil hayvan mağazası POS sistemi için farklı kullanıcı alt sınıfları şunları içerir:

- Kasiyerler
- Yöneticiler
- Sistem bakım personeli
- Mağaza müşterileri
- Envanter/depo personeli
- Muhasebeciler (vergi bilgilerini girmek için)
- Satış departmanı (fiyat ve indirim bilgilerini girmek için)

Bu kullanıcı alt gruplarının her birinin farklı istekleri vardır ve bunların belirlenmesi gerekir.

O zaman, ortaya çıkarmaya hazırlanma süreci şudur:

- Tüm müşterileri ve paydaşları belirleyin.
- Müşterileri ve diğer paydaş gruplarını ilgi alanlarına, kapsam, yetkilendirme veya diğer ayırt edici faktörlere göre sınıflara ayırın (bazı sınıflar birden fazla bölümlendirme düzeyine ihtiyaç duyabilir).
- Her kullanıcı sınıfı ve paydaş grubu için bir şampiyon veya temsili grup seçin.
- Her sınıftan veya paydaş grubundan ilk girdileri almak için uygun teknik(ler)i seçin.

İşte başka bir kullanıcı sınıfı bölümleme örneği. Bagaj taşıma sistemi için aşağıdakiler de dâhil olmak üzere birçok farklı paydaş vardır:

- Gezginler
- Sistem bakım personeli
- Bagaj görevlileri
- Havayolu planlayıcıları/dağıtıcıları
- Havalimanı personeli
- Havalimanı yöneticileri ve politika yapıcılar

Ancak her biri farklı ihtiyaçları olan çeşitli gezginler var. Örneğin, aşağıdaki alt sınıfları göz önünde bulundurun:

- Çocuklar
- Yaşlılar
- İş adamları
- Sıradan gezginler
- Askeri personel
- Siviller
- Sıradan gezginler
- Sık uçan yolcular

Bu alt sınıfların her birine farklı ortaya çıkarma teknikleri ile yaklaşılması gerekebilir. Örneğin, anketler çocuklar için uygun olmayabilirken, odak grupları askeri personel için daha az yararlı olabilir. Bu alt sınıfların çoğu örtüşür, örneğin bir kişi hem iş seyahatinde hem de sıradan bir gezgin olabilir ve ortaya çıkarma faaliyetlerinden elde edilen veriler analiz edilirken bu çakışmaların dikkate alınması gerekir.

ORTAYA ÇIKARMA TEKNİKLERİ ANKETİ

Şimdi ortaya çıkarma tekniklerini incelemeye başlama zamanı. Bu teknikleri alfabetik sırayla sunuyoruz; herhangi bir tercih ima edilmemektedir. Bölümün sonunda, bu tekniklerin farklı durumlarda yaygınlığını ve uygunluğunu tartışacağız.

Beyin Fırtınası

Beyin fırtınası, sistemler için kapsamlı hedefler oluşturmak için müşteriler ve diğer paydaşlarla yapılan resmi olmayan oturumlardan oluşur. Beyin fırtınası, belirlenmiş bir gündemi, tutanak tutmayı ve resmi yapıların kullanımını (örneğin, Robert'ın Düzen Kuralları) içerecek şekilde resmileştirilebilir. Ancak bir beyin fırtınası toplantısının formalitesi, muhtemelen toplantıda sergilenen yaratıcı seviyeyle ters orantılıdır. Bu tür toplantılar muhtemelen gayri resmi, hatta spontane olmalı ve herhangi bir büyük keşfin bazı kayıtlarını içeren tek yapı ile olmalıdır.

Beyin fırtınası oturumları sırasında bazı ön gereksinimler oluşturulabilir, ancak bu husus sürece ikincildir. JAD tekniği beyin fırtınasını (ve çok daha fazlasını) içerir ve muhtemelen diğer grup yönelimli ortaya çıkarma tekniklerinin çoğu, bir tür beyin fırtınasını örtük olarak somutlaştırır. Beyin fırtınası, misyon veya vizyon ifadesi oluşturma gibi genel hedef belirleme için de yararlıdır.

Kart Sıralama

Bu teknik, paydaşların sistem/yazılım ürünü için işlevsellik hakkında temel bilgileri içeren bir dizi kartı tamamlamasını içerir. Paydaşların/müşterilerin her bir işlevsellik için sıralama ve gerekçe içermesi de iyi bir fikirdir.

Müşterilerin ve paydaşların kartları tamamlamasına izin verecek süre önemli bir karardır. Kart sıralama alıştırması birkaç saat içinde tamamlanabilse de, paydaşların acele etmesi büyük olasılıkla önemli, eksik işlevlere yol açacaktır. Ancak paydaşlara çok fazla zaman vermek, süreci gereksiz yere yavaşlatabilir. Kartların tamamlanması için minimum 1 hafta (en fazla 2 hafta) süre verilmesi tavsiye

edilir. Diğer bir alternatif ise, müşterilerin kartları 2 saatlik bir oturumda tamamlamasını ve ardından 1 hafta sonra başka bir kart tamamlama ve inceleme oturumu için geri dönmesini sağlamaktır.

Her durumda, her kart oluşturma oturumundan sonra, gereksinim mühendisi bu kartları bir şekilde düzenler, genellikle işlevleri mantıksal olarak kümeler. Bu kümeler, belirlenen gereksinimlerin temelini oluşturur. Sıralanan kartlar, nihai kodda program sınıflarını belirlemek için CRC (yetenek, sorumluluk, işbirliği) kartları geliştirme sürecine girdi olarak da kullanılabilir. Kısaca tartışılacak başka bir teknik olan KFY, bir kart sıralama faaliyetini içerir.

Süreci açıklamak için, Şekil 3.1, müşteri tarafından evcil hayvan mağazası POS sistemi için oluşturulan ve sıralanmamış bir yığında yatan küçük bir kart alt kümesini göstermektedir.

Bu durumda, her kart, işlevselliğin yalnızca kısa bir açıklamasını içerir ve bir öncelik derecesi dâhildir (kısa olması için hiçbir gerekçe gösterilmemiştir).

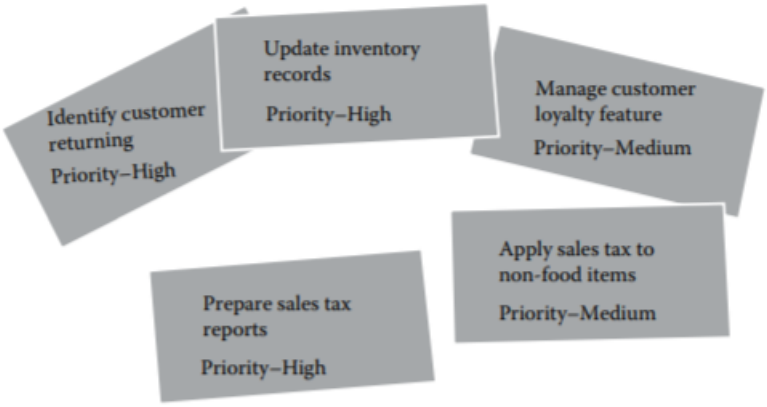


Figure 3.1 A tiny subset of the unsorted cards generated by customers for the pet store POS system.

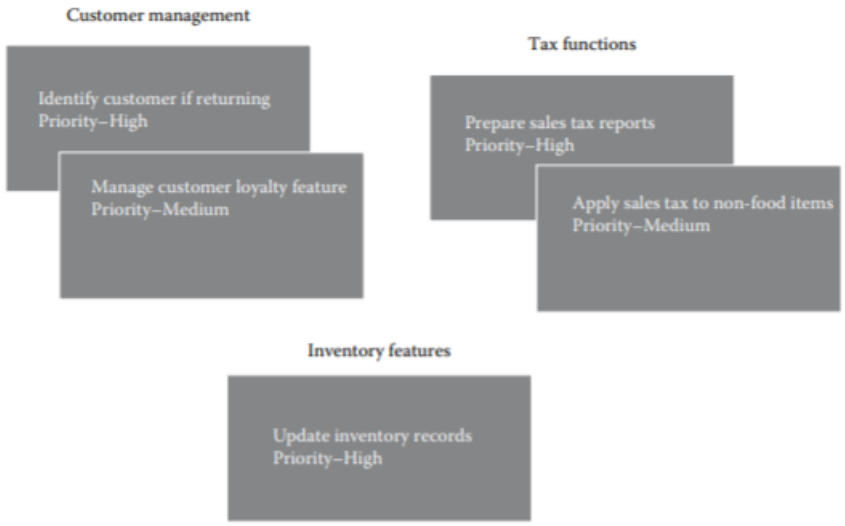


Figure 3.2 Sorted cards for the pet store POS system.

Gereksinim mühendisi bu kart yığınını analiz eder ve iki kartın "müşteri yönetimi" işlevlerine, iki kartın "vergi işlevlerine" ve bir kartın "envanter özellikleri" veya işlevlerine ait olduğuna karar verir ve kartları uygun yığınlar halinde düzenler. Şekil 3.2'de gösterilmiştir.

Müşteriye, düzeltme veya eksik özellikler için bu sıralanmış işlevler listesi gösterilebilir. Ardından, gerekirse yeni bir kart turu oluşturulabilir. Süreç, gereksinim mühendisi ve müşteri, sistem özelliklerinin büyük ölçüde yakalandığından emin olana kadar devam eder.

Çırak Olarak Tasarımcı

Çırak olarak tasarımcı, müşterinin ihtiyaçlarını anlamak için müşterinin işi hakkında yeterince bilgi edinmek için gereksinim mühendisinin müşterinin "omzunun üzerinden baktığı" bir gereksinim keşif tekniğidir. Müşteri ile tasarımcı arasındaki ilişki, usta ile çırak arasındaki ilişki gibidir. Yani, tıpkı gereksinim mühendisinin (tasarımcının) müşterinin işi hakkında müşteriden öğrenmesini istediğimiz gibi, çırak da ustadan bir beceri öğrenir. Çırak, ustanın bildiği her şeyi öğrenmek için oradadır (ve bu nedenle müşteriye işin bu kısımları hakkında konuşurken ve bunları gösterirken rehberlik etmelidir). Tasarımcı, belirli ihtiyaçları karşılamak için oradadır.

Müşterinin bu tekniğin çalışması için bir tür öğretme becerisine sahip olması gerekiyor gibi görünebilir, ancak bu doğru değil. Bazı müşteriler işleri hakkında etkili bir şekilde konuşamazlar, ancak ortaya çıktıkça onun hakkında konuşabilirler. Ayrıca, müşterilerin onu sunmanın en iyi yolunu veya güdülerini bulmaları gerekmez; sadece ne yaptıklarını açıklıyorlar.

Eseri görmek de neyin önemli olduğunu ortaya koyuyor. Örneğin, insanlar yaptıkları her şeyin ve bazen neden yaptıklarının farkında değildirler. Bazı eylemler yılların deneyiminin sonucudur ve ifade edilemeyecek kadar inceliklidir. Diğer eylemler, geçerli bir gerekçesi olmayan alışkanlıklardır. Bir çırağın mevcudiyeti, ustaya (müşteriye) faaliyetler ve nasıl ortaya çıktıkları hakkında düşünme fırsatı verir.

İşi görmek ayrıntıları ortaya çıkarır, çünkü bir görevi yerine getirmedikçe, onu tarif ederken detaylandırmak zordur. Son olarak, eseri görmek yapıyı ortaya çıkarır. Çalışma kalıpları işçi için her zaman açık değildir. Bir çırak, bir görevin birden fazla örneğini gözlemleyerek ve varyasyonları dâhil ederek, kendisinin nasıl yapılacağına dair bir anlayış oluşturarak işin stratejilerini ve tekniklerini öğrenir. Bu tekniğin çalışması için, gereksinim mühendisi aşağıdakiler dâhil işin yapısını ve anlamını anlamalıdır:

- İşi halletmek için strateji
- Yola çıkan kısıtlamalar
- Çalışmayı desteklediği için fiziksel ortamın yapısı
- İşin bölünmüş şekli
- Tekrarlayan aktivite kalıpları
- Bunların herhangi bir potansiyel sistem üzerindeki etkileri

Tasarımcı, herhangi bir yanlış anlamanın düzeltilebilmesi için müşteriye işi anladığını göstermelidir.

Son olarak, tasarımcıyı çırak yaklaşımı olarak kullanmak, gereksinimleri keşfetmenin ötesinde başka proje faydaları da sağlar. Örneğin, bu tekniği kullanmak, modellenmekte olan süreci iyileştirmeye yardımcı olabilir.

Hem müşteri hem de tasarımcı bu süreç boyunca öğrenir; müşteri neyin mümkün olabileceğini öğrenir ve tasarımcı da çalışma anlayışını genişletir. Ancak tasarımcının süreci iyileştirmek için bir fikri varsa, bu hemen (o anda) müşteriye geri gönderilmelidir.

Etki Alanı Analizi

Gereksinim mühendisliğinde alan bilgisine (ihtiyaç mühendisi ve/veya müşteri tarafından sahip olunmuş olsun) sahip olmanın önemini zaten vurgulamıştık. Etki alanı analizi, tasarlanan sistemle ilgili ve rekabet eden uygulamaların "manzarasını" değerlendirmeye yönelik herhangi bir genel yaklaşımı içerir. Böyle bir yaklaşım, temel işlevlerin ve daha sonra eksik işlevlerin belirlenmesinde faydalı olabilir. Etki alanı analizi daha sonra yeniden kullanılabilir bileşenleri (nihai tasarıma dâhil edilebilecek açık kaynaklı yazılım öğeleri gibi) belirlemek için de kullanılabilir. QFD ortaya çıkarma yaklaşımı açıkça alan analizini içerir ve bu tekniği kısaca tartışacağız.

Etnografik Gözlem

Etnografik gözlem, dolaylı ve doğrudan faktörlerin gözleminin gereksinim mühendisinin işini bilgilendirdiği herhangi bir tekniği ifade eder. Etnografik gözlem, sosyal bilimlerden ödünç alınan bir tekniktir; bu teknikte, insan faaliyeti ve işin meydana geldiği çevre gözlemlerinin, bilim adamını bazı fenomenlerin çalışmasında bilgilendirmek için kullanıldığı bir tekniktir. En katı anlamıyla, etnografik gözlem, uzun gözlem periyodlarını içerir (dolayısıyla, gerekleri ortaya çıkarma tekniği olarak kullanımına bir itiraz).

Etnografik gözlemi örneklemek için, farklı bir kültürü inceleyen antropologların toplumsal dalgınlığını hayal edin. Antropolog, incelenen kültür arasında yaşar, ancak bu, minimal düzeyde müdahalecidir. Kültür içinde yemek yerken, uyurken, avlanırken, kutlarken, yas tutarken vb. o toplumun nasıl işlediğine ve inanç sistemlerine dair doğrudan ve dolaylı her türlü kanıt toplanır.

Gereksinimlerin ortaya çıkarılmasına etnografik gözlem uygularken, gereksinim mühendisi müşterinin işyeri kültürüne kendini kaptırır. Burada, otomatikleştirilecek işi veya faaliyeti gözlemlemenin yanı sıra, gereksinim mühendisi ayrıca, doğrudan iletilemeyen, çevreden türetilen müşteri ihtiyaçlarına dair kanıt toplayabilecek bir konumdadır. Çırak olarak tasarımcı, etnografik gözlem etkinliğini içeren tek gereksinim ortaya çıkarma tekniğidir. Bu tekniği pratikte örneklemek için, etnografik gözlemin gerçekleştiği şu durumu göz önünde bulundurun:

- Bir müşteri için akıllı bir ev için gereksinimleri topluyorsunuz.
- Müşteriyle ne istedikleri hakkında uzun süre görüşerek zaman harcıyorsunuz.
- Müşteriler günlerini geçirirken ve sorular sorarken ("bulaşık makinesini neden gece çalıştırıyorsunuz, neden sabah çalıştırmıyorsunuz?") onlarla etkileşime geçerek zaman harcıyorsunuz.
- İstekler ve arzular hakkında sözsüz ipuçları almak için müşteriyi mevcut evde “hareket halindeyken” pasif bir şekilde gözlemleyerek uzun zaman harcarsınız.
- Kitaplıktaki kitaplar, duvardaki tablolar, mobilya stilleri, hobilerin kanıtları, çeşitli cihazlarda eskime ve yıpranma belirtileri gibi diğer bilgileri evin kendisinden edinirsiniz.

Etnografik gözlem çok zaman alıcı olabilir ve gözlemcinin önemli ölçüde eğitim almasını gerektirir. Sürecin müdahaleciliğine dayanan başka bir itiraz daha var. Fizikte, Heisenberg belirsizlik ilkesi olarak bilinen iyi bilinen bir ilke vardır; bu, sıradan kişilerin terimleriyle, ölçtüğünüz şeyi etkilemeden bir şeyi tam olarak ölçemeyeceğiniz anlamına gelir. Örneğin, müşteri için çalışma

ortamını gözlemlerken, herkes etkilemek için dışarı çıktığı için süreçler ve davranışlar değişir - bu nedenle durumun yanlış bir resmi oluşturulur ve hatalı kararlara yol açar.

Hedefe Dayalı Yaklaşımlar

- Hedefe dayalı yaklaşımlar, gereksinimlerin, bir dizi hedef aracılığıyla misyon bildirimi kaynaklı bir ortaya çıkarma tekniği ile gerçekleşir. Misyon bildiriminde bir dizi hedef oluşturulur. Bu hedefler, bir veya daha fazla kez alt bölümlere ayrılır. Daha sonra, alt düzey hedefler belirli üst düzey gereksinimlere ayrılır. Son olarak, üst düzey gereksinimler daha düşük düzeyli gereksinimler oluşturmak için kullanılır.

- Örneğin, bagaj taşıma sistemine misyon bildirimi ile bakalım:

Amaç: Yolcu kalkış noktasından varış noktasına kadar bagaj taşımanın tüm yönlerini otomatikleştirmek.

- Hedef 1: Check-in'den teslim alınmasına kadar bagaj takibini tamamen otomatik hale getirmek.

- Hedef 2: Bagajın check-in kontuarından uçağa yönlendirilmesini tamamen otomatikleştirmek.

- Hedef 3: Kayıp bagaj miktarını %1'e düşürmek.

- Bu hedefler daha sonra, hedef-soru-metrik (GQM) gibi yapılandırılmış bir yaklaşım kullanılarak gereksinimlere ayrıştırılır. GQM, gereksinim sistem mühendisliğinin birçok alanında kullanılan önemli bir tekniktir. GQM üç adımı içerir: sistemin amaçlarını veya hedeflerini belirtmek; hedeften hedefe ulaşılıp ulaşılmadığına dair sorular türetmek; soruları cevaplayabilmek için neyin ölçülmesi gerektiğine karar vermek. Örneğin, bagaj taşıma sistemi durumunda, hedef 3'ü düşünün. belirli bir (havaalanı/havayolu/uçuş/zaman aralığı/vb.)?" Bu soru, formun gerekliliği olduğunu öne sürüyor:

- Belirli bir [havaalanı/havayolu/uçuş/zaman aralığı/vb.] için kaybolan bagaj yüzdesi, %1'den fazla olmayacaktır.

    - Burada bilerek basit bir örnek seçtik .Normalde problemler her zaman çok açık değildir ve ilgili ölçüm de her zaman çok açık olmaz.Bu tür durumlarda GQM devreye girmelidir.

Örneğin aşağıdaki gereksinime bakalım:

Sistem kullanıcı dostu olmalıdır.

Böyle bir gereksinimle ilgili sorun, memnuniyetini göstermenin bir yolu olmamasıdır. GQM'yi takiben, kullanıcının amacına ilişkin bir dizi soru oluşturuyoruz:

1. Sistemi kullanmayı öğrenmek ne kadar kolay?

2. Yeni bir kullanıcının ne kadar yardıma ihtiyacı var?

3. Bir kullanıcı kaç hata alıyor?

Ardından, bu soruların her biri için bir veya daha fazla metrik tanımlıyoruz.

1. Bir kullanıcının belirli işlevleri nasıl gerçekleştireceğini öğrenmesi için geçen süre..

2. Bir kullanıcının belirli bir süre boyunca yardım özelliğini kaç kez kullanması gerektiği..

3. Bir kullanıcının belirli bir süre boyunca belirli işlemler sırasında bir hata mesajı görme sayısı..

Son olarak, bu metrikler için kabul edilebilir aralıklar belirlemek üzere müşteriyle birlikte çalışırız. Sonrasında sistem inşa edildiğinde, gerçek kullanıcılarla yapılan test denemeleri yoluyla gereksinimlerin tatmini gösterilebilir. Bu şekilde kabul edilebilir bir kullanım kolaylığı seviyesi tanımlayabiliriz.

Grup Çalışması

- ❑ Grup çalışması, ihtiyaç keşfi, analizi ve takibi süreçlerinde kullanılan her türlü grup toplantılarının genel adıdır. Gereksinimlerin ortaya çıkarılması için grup odaklı çalışmaların en ünlüsü, birazdan tartışacağımız ortak uygulama tasarımıdır (JAD).

Grup faaliyetleri, birçok paydaşı bir araya getirme açısından çok verimli olabilir, ancak çatışma ve bölünme potansiyelini riske atabilir. Her türlü grup çalışmasında başarının anahtarı, grup toplantılarının planlanması ve yürütülmesidir. İşte grup toplantıları hakkında hatırlanması gereken şeyler

- Toplantı gerçekleşmeden birkaç gün önce bir gündem yayınlayın
- Toplantı boyunca gündemde kalın.
- Elinizde özel bir not alıcı bulundurun.
- Kişisel sorunların araya girmesine izin vermeyin.
- Herkesin sesini duyurmasına izin verin.
- İlk fırsatta fikir birliği arayın.
- Gündemdeki tüm maddeler yeterince tartışılana kadar ayrılmayın.
- Toplantı tutanaklarını yayınlayın ve katılımcıların değişiklik önermesine izin verin.

Bu ilkeler, JAD ‘ı ortaya çıkarır.

- ❑ Grup çalışmasının birçok dezavantajı vardır. İlk olarak, grup toplantılarını organize etmek ve dahil olan paydaşlarının konulara odaklanmasını sağlamak zor olabilir.
- ❑ Tüm katılımcılar kendini rahat bir şekilde ifade edemeyebilir. Bundan dolayı bazı kişiler çok aktif olabilirken bazıları pasif kalabilir. Toplantıları yürütmek ve bunu bir grup ile yapmak, gelişmiş liderlik ve çeşitli beceriler gerektirir. Gereksinim mühendisi bunları yapacak kapasiteye sahip olmalıdır.

Mülakatlar

Mülakatlar yoluyla ortaya çıkarma, iki kişi arasındaki yüz yüze iletişimi Sistem düzeyinde gereksinimleri ortaya çıkarmak için kullanımı kolay bir tekniktir. Üç gruba ayrılır ve bireylere veya odak gruplarına uygulanabilirler:

- Yapılandırılmamış, Yapılandırılmış, Yarı yapılandırılmış
  - Yapılandırılmamış görüşmeler, konuşmacı ve katılımcıları rahatlatmaya yöneliktir. Herhangi bir zamanda ve herhangi bir yerde ortaya çıkabilir. Mühendis ve paydaş bir aradadır bu sebeple kesintisiz bilgi edinme fırsatı sağlanır. Fakat çok tercih edilmemelidir.
  - Yapılandırılmış görüşmeler, çok daha resmidir ve önceden tanımlanmış, planlanmış sorulardır. Şablonlar kullanılır. Bu türün dezavantajı çok kontrollü olmasından dolayı bazı müşteriler bilgi saklayabilirler.
  - Yarı yapılandırılmış görüşmeler, yukarıdaki ikisinin en iyilerini birleştirir. Yani, gereksinim mühendisi dikkatlice düşünülmüş bir soru listesi hazırlar, ancak daha sonra görüşme sırasında ekstra soruların da içeri girmesine izin verir.
  - Farklı organizasyon yapılarına göre hangi türün kullanılacağı değişiklik gösterir.

Örnek bazı mülakat sorularına bakalım:

- Sistemin önemli bir özelliğini adlandırın.
- Bu özellik neden önemli?
- Bu özellik diğer özelliklere göre ne kadar önemlidir?
- Başka hangi özellikler bu özellikten bağımsız olmalıdır?
- Bu özellik hakkında başka ne gibi gözlemler yapabilirsiniz?

Hangi görüşme tekniği kullanılırsa kullanılsın, tüm doğru soruların sorulduğundan emin olmak için özen gösterilmelidir. Gerekli olduğunda telefon, video konferans veya e-posta yoluyla görüşmeler yapılabilir.

İç Gözlem

Bir gereksinim mühendisi, gereksinimleri ne düşündüğüne dayalı olarak geliştirdiğinde, müşteri isterse, o iç gözlem sürecini yürütür.

Özünde, gereksinim mühendisi kendini müşterinin yerine koyar ve “müşteri ben olsaydım sistemin bunu yapmasını isterdim…” diye düşünür. Gereksinim mühendisinin alan bilgisi müşterininkinden çok daha fazla olduğundan, içe dönük bir yaklaşım yararlıdır. Bazen müşteri, mühendise şöyle sorabilir: "Ben olsan ne isterdin?«

İç gözlem, gereksinim mühendisinin empati kurarak bu tür soruları cevaplandırmasını sağlar.

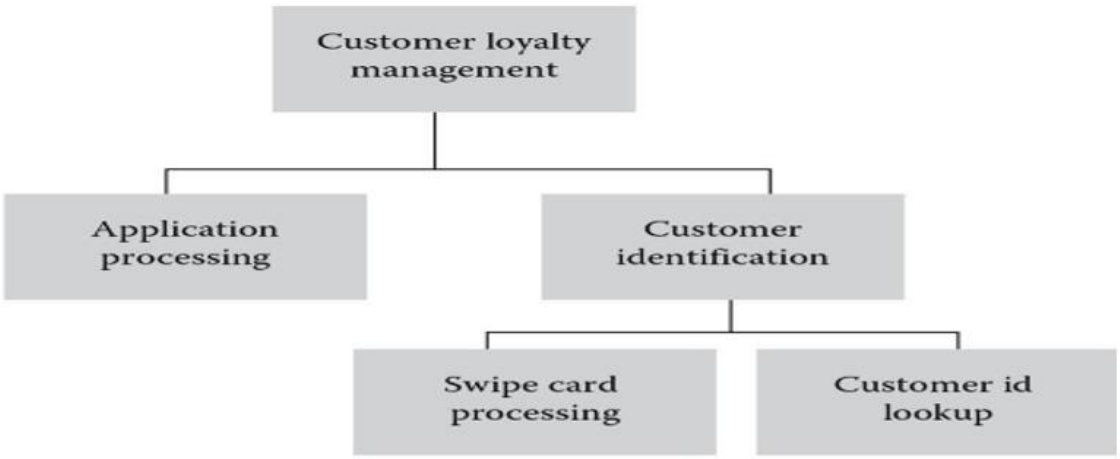
Ortak Uygulama Tasarımı

- Ortak uygulama cihazı (JAD), sistem kullanıcıları, sistem sahipleri ve uzun bir süre boyunca belirli bir dizi soruna odaklanan analistler ile yüksek düzeyde yapılandırılmış grup toplantıları içerir. Bu toplantılar günde 4-8 saat ve 1 günden birkaç haftaya kadar süren bir süre içinde gerçekleşir. Geleneksel olarak büyük hükümet sistemleri projeleriyle ilişkilendirilse de teknik, endüstriyel ortamlarda her boyuttaki sistemlerde kullanılabilir. Spesifik olarak, gereksinim mühendisi, operasyon tanımı kavramı, sistem hedefi tanımı, gereksinimlerin ortaya çıkarılması, gereksinim analizi, gereksinim belgesi incelemesi ve daha fazlası için JAD oturumlarını kullanabilir.
- Bir JAD incelemesi veya denetim oturumu planlaması üç adımdan oluşur:
- 1. Katılımcıların seçilmesi
- 2. Gündemin hazırlanması
- 3. Bir yer seçme
- Bu adımların her birini hazırlarken büyük özen gösterilmelidir.İncelemeler ve denetimler, aşağıdaki katılımcıların bir kısmını veya tamamını içerebilir:
- ▪ Sponsorlar (ör. üst yönetim)
- ▪ Bir ekip lideri (kolaylaştırıcı, bağımsız)
- ▪ Gereksinimlerin ve iş kurallarının sahibi olan kullanıcılar ve yöneticiler
- ▪ Katipler (yani toplantı tutanakları ve not tutanlar)
- ▪ Mühendislik personeli
- Bir oturum planlamadan önce, analist ve sponsor projenin kapsamını belirlemeli ve her oturumun üst düzey gereksinimlerini ve beklentilerini Gündem büyük ölçüde yürütülecek incelemenin türüne bağlıdır ve yeterli zamana izin verecek şekilde oluşturulmalıdır.
- Aşağıda yazılım gereksinimleri, tasarım denetimleri veya kod adım adım ilerlemek için bazı kurallar yer almaktadır. Oturum lideri, bu uygulamaların uygulanmasını sağlamak için her türlü çabayı göstermelidir.
- ▪ Gündeme bağlı kalın.
- ▪ Programa bağlı kalın (gündem konularına belirli bir zaman ayrılır).
- ▪ Yazıcının not alabildiğinden emin olun.
- ▪ Teknik jargondan kaçının (inceleme teknik olmayan personeli içeriyorsa).
- ▪ Çatışmaları çözün (onları ertelememeye çalışın).
- ▪ Grup fikir birliğini teşvik edin.

- ▪ Kişilerin oturuma hakim olmasına izin vermeden kullanıcı ve yönetim katılımını teşvik edin.
- ▪ Toplantıyı kişisel olmayan tutun.
- ▪ Toplantıların gerektiği kadar uzun sürmesine izin verin
- Herhangi bir gözden geçirme oturumunun son ürünü, tipik olarak, oturum sırasında üzerinde anlaşmaya varılan öğelerin (spesifikasyonlar, tasarım değişiklikleri, kod değişiklikleri ve eylem öğeleri) bir özetini sağlayan resmi bir yazılı belgedir. Belgenin içeriği ve organizasyonu, açık bir şekilde, oturumun doğasına ve amaçlarına bağlıdır. Ancak gereksinimlerin ortaya çıkması durumunda, ana eser SRS'nin ilk taslağı olabilir.

Merdivenleme

Merdivenlemede, gereksinim mühendisi, gereksinimleri ortaya çıkarmak için müşteriye kısa yönlendirici sorular (sondalar) sorar. Yanıtlardan elde edilen bilgiler daha sonra ağaç benzeri bir yapı halinde düzenlenir.



Laddering diagram for the pet store POS system.

Örneğin, evcil hayvan mağazası POS sistemi için aşağıdaki merdiven sorularını ve yanıtlarını göz önünde bulundurun. “RE”, gereksinim mühendisini ifade etsin:

RE: Sistemin önemli bir özelliğini adlandırın.

Müşteri: Müşteri kimliği.

RE: Bir müşteriyi nasıl tanımlarsınız?

Müşteri: Sadakat kartlarını okutabilirler.

RE: Bir müşteri kartını unutursa ne olur?

Müşteri: Telefon numarasından aranabilirler.

RE: Müşterinin telefon numarasını ne zaman alırsınız?

Müşteri: Müşteriler sadakat kartı başvurusunu tamamladığında.

RE: Müşteriler başvuruları nasıl tamamlıyor? …

Ve bunun gibi…

Protokol Analizi

Protokol analizinde, müşterileri gereksinimleri ile mühendisler, otomatikleştirecekleri prosedürleri gözden geçirir. Böyle bir gözden geçirme sırasında müşteriler, atılan her adımın gerekçesini açıkça belirtir. Mühendisler tarafından sahada yapılan gözlemler genellikle süreç optimizasyonuna ve diğer yeniliklere yol açar. Protokol analistinin çırak olarak tasarımcıya çok benzediği görülür, ancak ince farklılıklar vardır. Bu farklılıklar, çırak olarak tasarımcıdan ziyade protokol analizinde daha pasif olan gereksinim mühendisinin rolünde yatmaktadır.

Prototipleme

Prototipleme, yenilerini keşfetmek için sistem modellerinin ve özellikle kullanılabilirlik gereksinimlerinin oluşturulmasıdır. Prototipleme, gereksinimleri ortaya çıkarmak için özellikle önemli bir tekniktir. Örneğin, spiral yazılım geliştirme modeli ve çevik metodolojiler yaygın olarak kullanılır. Temelinde bir dizi giderek işlevsel olmayan atılabilir prototipten oluşur. Prototipler, çalışan modelleri ve çalışmayan modelleri içerebilir. Çalışan modeller, yazılım sistemleri ve simülasyonlar söz konusu olduğunda çalıştırılabilir. Kod veya yazılım dışı sistemler için geçici veya ölçekli prototipler içerebilir. Çalışmayan modellerde kullanıcı arayüzlerinin storyboard'larını ve maketlerini içerebilir. Bina mimarlarının müşteri gereksinimlerinin ortaya çıkarılmasına ve onaylanmasına yardımcı olmak için düzenli olarak prototipler kullanırlar. (örn. ölçekli çizimler, karton modeller, 3 boyutlu bilgisayar animasyonları). Aynı sebeplerden dolayı sistem mühendisleri de prototipleri kullanır.

Çalışan yazılım prototipleri durumunda, kod kasıtlı olarak atılmak üzere(ıskarta etmek için) tasarlanmıştır veya kasıtlı olarak yeniden kullanılmak üzere tasarlanabilir (atılır atılmaz)

Örneğin, grafik kullanıcı arabirimi kod maketleri gereksinimleri ortaya çıkarmak için tam olarak kullanılabilir ve kod yeniden kullanılabilir. Ve çevik yazılım geliştirme metodolojileri, atılmayan prototipleri sürekli olarak geliştirme sürecini içerir. Ve çevik yazılım geliştirme metodolojileri, sürekli gelişen, atılmayan prototipler sürecini içerir.

Son zamanlarda, 3 boyutlu baskı, fiziksel yapı oluşturmada önemli bir araç haline geldi. Hızlı prototipleme teknolojilerinden 3 boyutlu baskının diğerlerine göre iki önemli avantajı vardır. Birincisi maliyettir—endüstriyel kalite 3 boyutlu yazıcılarla sağlanır. Hızlı prototipleme makineleri birkaç bin dolara satın alınabilirken geleneksel bilgisayar sayısal kontrolünü (CNC) kullanmak yüzlerce dolara mal olabilir. İkinci avantaj, 3 boyutlu yazıcıların girdi standardı olarak alabilmesidir. Yaygın olarak kullanılan bilgisayar destekli tasarım (CAD) programları tarafından üretilen dosyaları biçimlendirir. (Berman 2012).

Prototiplemeyi kullanmanın birkaç farklı yolu vardır; örneğin, içinde dördüncü nesil bir ortam (yani bir simülatör), atılabilir prototipleme, evolutionary prototipleme (prototipin son sisteme evrildiği yer) veya kullanıcı arayüz prototipleme. Bazı kuruluşlar, birden fazla ön tipleme türü kullanabilir.

Kassab et al. (2014) yazılım uzmanlarına yönelik gereksinim mühendisliği ile ilgili çeşitli uygulamaların yaygınlığını belirleyen geniş bir anket gerçekleştirmiştir. Bu farklı prototipleme için seçim sıklığına ilişkin sonuçları teknikler Şekil 3.4'te gösterilmiştir.

Gereksinimlerin ortaya çıkarılması için prototiplemeyi kullanırken göz önünde bulundurulması gereken en az üç tehlike vardır.

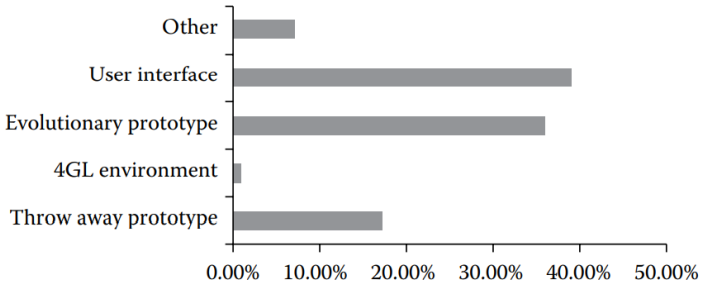


Figure 3.4 Prototype methods selection across software development life cycle methodology. (Adapted from Kassab, M., Neill, C & Laplante, P. State of practice in requirements engineering: contemporary data. *Innovations in Systems and Software Engineering*, 10, no. 4 (2014): 235–241. With permission.)

İlk olarak, bazı durumlarda, saklanması amaçlanmayan yazılım prototipleri aslında program baskıları nedeniyle tutulur. Bu durum potansiyel olarak

tehlikelidir, çünkü kod muhtemelen en titiz teknikler kullanılarak tasarlanmamıştır.

Kullanılıp atılan prototiplerin istenmeyen yeniden kullanımı, endüstride sıklıkla meydana gelir. İkinci sorun, prototiplemenin bazı işlevsel olmayan gereksinimlerle keşifte her zaman etkili olmamasıdır.

Bu, özellikle yalnızca geçerli standartların ve yasaların analizi ile elde edilebilecek şartlar gerektirenler için geçerlidir. (Kassab ve Ormancıeva 2014)

Son olarak, yöntemleri keşfetmek için prototipleri kullanırken hangi kullanıcıların sistemle etkileşime girdiği gibi sorunlar ortaya çıkabilir.

Asıl endişe, gerçek sisteme karşı kullanıcıların bir prototiple (davranışın sonuçlarının gerçek olmadığı) farklı bir şekilde etkileşime girmesidir.

Örneğin, kullanıcıların bir araçta nasıl araç kullanabileceğini düşünün. Bir kazadan kaynaklanan gerçek yaralanma veya hasarın olmadığı simülatör. Sürücüler simülatörde gerçek bir araçta olduğundan çok daha agresif davranabilir. Bu da hatalı gereksinimlerin tespit edilmesine yol açar.

KALİTE FONKSİYON DEĞİŞİMİ

Kalite işlev dağıtımı (QFD), müşteriyi keşfetmeye yönelik bir tekniktir . QFD, müşterilerin ihtiyaçlarının karşılanması için bir yapı sağlar.

İstekler dikkatlice dile getirilir daha sonra teknik gereksinimler doğrudan bir şirketin iç kısmına yönlendirilir. Analizden uygulamaya ve dağıtıma kadar bu devam eder.

QFD'nin temel fikri, müşteri arasındaki ilişki matrislerini oluşturmaktır, ihtiyaçlar, teknik gereksinimler, öncelikler ve (gerekirse) rakip değerlendirmesi gibi. Özünde, QFD, kart sıralama, merdivenleme ve alan analizini içerir. Bu ilişki matrisleri genellikle bir evin çatısı, tavanı ve yanları olarak temsil edildiğinden, QFD bazen “kalite evi” olarak anılır. (Figure 3.5; Akao 1990).

QFD, 1966 yılında Yoji Akao tarafından ağır imalatta kullanılmak üzere tanıtıldı. Endüstri ve sistem mühendisliği tarafından yazılım sistemlerine de uygulanmıştır.

IBM, DEC, HP, AT&T, Texas Instruments ve diğerleri.

"Müşterinin sesi" derken, mühendis, müşterilerin neye ihtiyaç duyduklarını, gereksinimlerini anlamak için empatik bir şekilde müşterileri dinlemelidir.

Müşterinin sesi tüm analiz, tasarım ve geliştirme faaliyetlerinin temelini oluşturur; ürünler sadece "mühendisin sesinden" geliştirilmez." Bu yaklaşım gereksinimlerin ortaya çıkarılmasının özüdür.

Aşağıdaki gereksinim mühendisliği süreci, QFD tarafından öngörülmüştür.

- Paydaşların niteliklerini veya gereksinimlerini belirleyin.
- Gereksinimlerin teknik özelliklerini belirleyin.
- Gereksinimleri teknik özelliklerle ilişkilendirin.
- Rakip ürünlerin bir değerlendirmesini yapın.
- Teknik özellikleri değerlendirin ve her özellik için bir hedef değer belirleyin.
- Geliştirme çalışmaları için teknik özelliklere öncelik verin.

QFD, rekabetçi analize yönelik bir yaklaşım kullanır.

Yani, bir özellik listesi rekabetçi ürünler için ilgili tüm özelliklerin birleşiminden oluşturulur. Bu özellikler bir rekabet matrisinin sütunlarını içerir. Satırlar aşağıdakileri temsil eder:

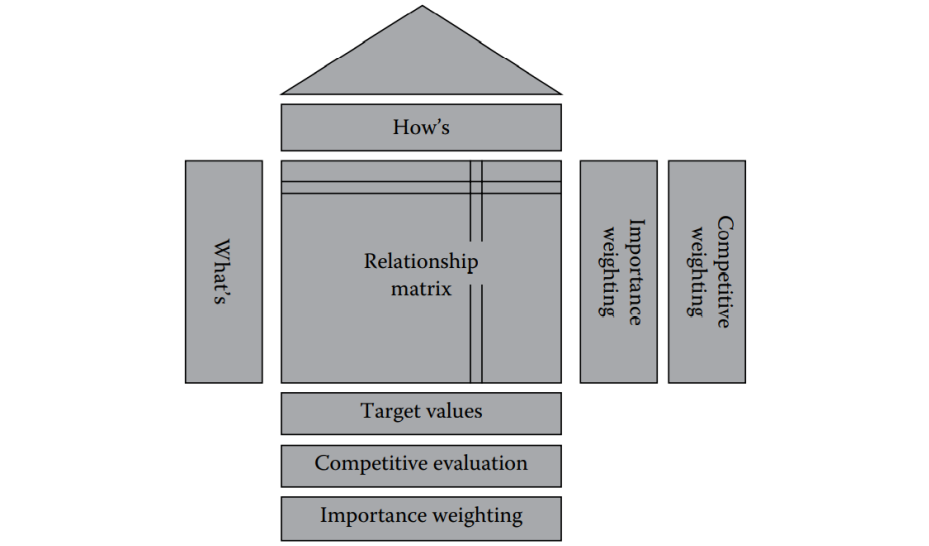


Figure 3.5 QFD's "house of quality" (Akao 1990).

Rakip ürünler ve ilgili hücreler bu özellikler için doldurulur bu her ürün için yapılır.

Matris daha sonra bir başlangıç kümesini, yeni veya revize edilmiş ürün için gereksinimleri formüle etmek için kullanılabilir. Matris ayrıca aşağıdakileri sağlamaya yardımcı olur: Temel özelliklerin yeni sistemde atlanmaması ve iyileştirmeye katkıda bulunabilir, istenen kalite gereksinimlerinin tamlığına yardımcı olur.

Örneklemek gerekirse, evcil hayvan mağazası POS sistemi için kısmi bir rekabet analizi:

Tablo 3.1'de gösterilmiştir.

Detaylandırabilmemize rağmen, yalnızca çok üst düzey özelliklerin gösterildiğine dikkat edin.

İstenilen ayrıntı seviyesine kadar, matrisi büyük ölçüde genişletir. Matris bize zorunlu ve isteğe bağlı özelliklerden oluşan bir başlangıç seti sunar.

Örneğin, tüm bu ürünlerde kablosuz destek özelliği bulunur, bu özelliğin yeni evcil hayvan mağazası POS sisteminde bulunduğunu belirtmek zorunludur.

Gereksinim mühendisliğinde toplam yaşam döngüsü yaklaşımı içerdiğinden,

Qfd'nin diğer bağımsız çıkarma tekniklerine göre birçok avantajı vardır. QFD kullanıcıların ve yöneticilerin katılımını artırır. Gelişimi kısaltır yaşam döngüsü ve genel proje geliştirmeyi iyileştirir. QFD, iletişim süreçlerini yapılandırarak ekip katılımını destekler. Son olarak, bilgi kaybını önleyen bir araçtır.

Bununla birlikte, QFD'nin bazı dezavantajları vardır. Örneğin, zamansal gereksinimleri ifade etmede zorluklar olabilir. Ve QFD'nin kullanımı zordur.

Yeni bir proje türü için müşteri gereksinimlerini nasıl keşfedersiniz?

Rekabet gücünü nasıl inşa eder ve analiz edersiniz?

Bu durumlarda, çözüm benzer veya ilgili ürünlere bakmaktır, ancak yine de bilişsel bir boşluk olması muhtemeldir.

Table 3.1 Partial Competitive Analysis for Pet Store Point of Sale System

	Competing Product		
Feature	*MyFavoritePet*	*BestFriends*	*Fido-2.0*
Maximum simultaneous users supported	100	250	Unlimited
Wireless device support	Yes	Yes	Yes
Business analytics features	Yes	Yes	No
Operating system support	Windows/ Mac/Linux	Windows/Linux	Windows/Mac
Cost (base system) ($K)	50	110	75

Gereksinimlerin Ortaya Çıkarılması

Bazen belirli işlevler için ölçümleri bulmak ve bunları tutmak zordur. Ve ne kadar az bilirsek, o kadar az belgeleriz.

Son olarak, özellik listesi kontrolsüz bir şekilde büyüdükçe, kalite evi bir "Konak."

QFD, birincil gereksinim ortaya çıkarma yaklaşımı olarak kullanılmasa bile, mümkün olan her yerde rekabetçi sistem analizine yaklaşımı kullanılmalıdır. QFD rekabet analizinin yapılandırılmış yapısı, aşağıdakileri sağlamanın etkili bir yoludur. Önemli gereksinimlerin eksik olmaması, daha eksiksiz bir gereksinim kümesi sağlar.

Anketler

Gereksinim mühendisleri genellikle geniş paydaş gruplarına ulaşmak için anketleri ve diğer anket araçlarını kullanır.

Anketler genellikle sürecin erken aşamalarında kullanılır.

Kapsam sınırlarını hızlı bir şekilde tanımlamak için bir seçim sürecidir. Her türden anket sorusu kullanılabilir. Örneğin, kapalı uçlu (örneğin, çoktan seçmeli, doğru-yanlış) veya açık uçlu—serbest biçimli yanıtları içerir.

Kapalı sorular, analiz için daha kolay kodlama avantajına sahiptir ve sistemin kapsamını bağlar.

Açık sorular daha fazla özgürlük ve yenilik sağlar, ancak analiz edilmesi daha zor olabilir ve kapsam kaymasını teşvik edebilir.

Örneğin, evcil hayvan mağazası POS sistemi için olası bazı anket soruları şunlardır:

Envanterinizde kaç tane benzersiz ürün (SKU) taşıyorsunuz?

(a) 0–1000; (b) 1001–10,000; (c) 10.001–100.000; (d) >100.000

- Kaç farklı depo siteniz var? ____
- Kaç farklı mağaza konumunuz var? ____
- Şu anda kaç benzersiz müşteriniz var? ____

Sorular, kapalı uçlu sorular için bile yeterince çerçevelenmemişse, fazla kapsam ve alt kapsam belirleme tehlikesi vardır.

Bu nedenle anket çalışması teknikler, alan her ikisi tarafından (paydaşlar ve gereksinim mühendisi) da çok iyi anlaşıldığında en kullanışlıdır.

Büyük ölçekli anketler yapmadan önce amaçlanan anket popülasyonunun küçük bir alt kümesiyle bir pilot çalışma yapmak önemlidir.

Sonuçlar analiz edilir ve kafa karıştırıcı, eksik veya konu dışı soruları belirlemek amacıyla anket katılımcıları ile görüşülmüştür.

Daha sonra, anket daha büyük bir nüfusa uygulanmadan önce araç rafine edilebilir.

Anket verilerini analiz ederken, özellikle katılımcılardan kimliklerini belirlemeleri istenir aşağıdaki etkiye dikkat edin.

Gerçekleştirilmesi gerekmeyen bir dizi seçenek verildiğinde, kişi karar verildiyse çok daha fazla sayıda seçeneği arzu etme eğiliminde aslında yapılacak.

Yazılım ve Sistemler için Gereksinim Mühendisliği

Aşağıdaki örnekten dolayı bu etkiyi “dondurma deposu etkisi” olarak adlandırıyoruz.

El yapımı bir dondurma dükkânı açmaya karar veren bir girişimci düşünün.

Ürün araştırmasının bir parçası olarak, birkaç kişiyle anket yapıyor.

Ankete katılanlar, satın alacakları dondurmanın lezzetlerini kontrol ediyorlar.

Ankette listelenen 30 farklı lezzetten 20'sinin ankete katılanların %50'si veya daha fazlası tarafından seçiliyor.

Böylece üretici üretmeye karar veriyor ve bu 20 aromayı kabaca ankette belirtilen taleple orantılı olarak stoklar.

Sonuçlar.

Yine de, dondurma dükkânını açtıktan 1 hafta sonra, 90% çikolata, vanilya ve çilek - onun iş ilk üç tatlar kaynaklanmaktadır. Envanterinde tuttuğu anketteki diğer 17 lezzetten birkaçı daha önce hiç tadılmadı. Müşterilerin satın alacaklarını söylemelerine rağmen ankette bu tatlar, seçimlerini yapma zamanı geldiğinde farklı davrandılar

Bu nedenle, müşterilerleyken dondurma dükkânının etkisini unutmayın.

Özellik kümeleriyle ilgili seçimlerde —bir şey söyleyip başka bir şey yapacaklar.

Anketler telefon, e-posta, yüz yüze ve Web tabanlı olarak gerçekleştirilebilir.

Anket oluşturma sürecini basitleştirmek ve kullanılması gereken sonuçları toplamak ve analiz etmek için kullanılabilir çeşitli ticari araçlar ve açık kaynaklı çözümler vardır.

RAPOR IZGARALARI (REPORTY GRİDS)

Repertuar ızgaraları, çeşitli özellikler için yapılandırılmış bir sıralama sistemi içerir. Sistemde farklı varlıklar bulunur.

Repertuar ızgaraları, özellikle paydaş grupları içindeki anlaşmazlıkların belirlenmesi için yararlıdır.

Izgaralar, satırların sistemi temsil ettiği bir özellik veya kalite matrisi gibi görünür.

Varlıklar ve istenen nitelikler sütunları, paydaşlar da her birine dayalı sıralamaları temsil eder.

Izgaralar hem nitelikleri hem de özellikleri bünyesinde barındırabilirken, genellikle şebekelerin, analiz ve uyuşmazlık çözümünün tutarlılığını sağlamak için tüm özelliklere veya tüm niteliklere sahip olması durumudur.

Tekniği göstermek için, Şekil 3.6 çeşitli için bir repertuar ızgarasını temsil eder. Bagaj taşıma sisteminin nitelikleri

Burada havalimanı operasyonları için görüyoruz yönetici, tüm nitelikler esasen en yüksek öneme sahiptir (güvenlik biraz daha düşük, 4).

Baggage handling speed	1	1	5
Fault-tolerance	4	5	5
Safety	5	4	4
Reliability	3	5	5
Ease of maintenance	3	5	5

Airport operations manager
Maintenance engineer
Airline worker's union rep

Figure 3.6 Partial repertory grid for the baggage handling system.

Ancak Havayolu İşçileri Sendikası temsilcisi için güvenlik en önemli şeydir (sonuçta sendika üyeliğinin sistemle belirli bir düzeyde etkileşime girmesi gerekir).

Özünde, bu derecelendirmeler, paydaşların gündemlerini veya farklı bakış açılarını yansıtır. Bu nedenle, repertuar ızgaralarının paydaş hedeflerini içeren anlaşmazlıklarla erken yüzleşmede kullanımının yararlı olduğunu görürüz.

Ayrıca, ızgaralar daha sonra anlaşmazlıklarla başa çıkmak için değerli belgeler sağlayabilir

Yine de, repertuvar ızgaralarını kullanırken, dondurma dükkanının etkisini hatırlayın—paydaşlar bir şey söyleyecektir ve daha sonra farklı davranır.

SENARYOLAR

Senaryolar, kullanımda olan sistemin üst düzey bir performans sağlayan sistem çalışmasının tanımı, kullanıcı sınıfları ve istisnai durumlar gibi resmi olmayan açıklamalarıdır.

İşte pet shop POS sistemi için örnek bir senaryo.

Bir müşteri evcil hayvan mağazasına girer ve arabayı çeşitli ürünlerle doldurur. Kontrol ederken, kasiyer müşterinin bir sadakat kartı olup olmadığını sorar. Bu durumda, kasiyer kartı okutarak müşterinin kimliğini doğrular. Değilse, kasiyer kartı yerinde tamamlamayı teklif eder.

Sadakat kartı faaliyetinden sonra kasiyer, ürünleri bir barkod okuyucu kullanarak tarar. Her kalem tarandıkça, satış toplanır ve envanter uygun şekilde güncellenir. Ürün taraması tamamlandıktan sonra bir ara toplam hesaplanır. Ardından kupon ve indirimler girilir. Yeni bir ara toplam hesaplanır ve geçerli vergiler eklenir. Bir makbuz yazdırılır ve müşteri nakit, kredi kartı, banka kartı veya çek kullanarak ödeme yapar. Tüm uygun toplamlar (satışlar, vergiler, indirimler, indirimler vb.) hesaplanır ve kaydedilir.

Etki alanı yeni olduğunda senaryolar oldukça kullanışlıdır (etki alanı için bir senaryo düşünün uzay istasyonu, örneğin). Kullanıcı hikâyeleri aslında bir senaryo biçimidir.

GÖREV ANALİZİ

Hâlihazırda incelemiş olduğumuz hiyerarşik yönelimli tekniklerin çoğu gibi, görev analizi, ilgili kişi tarafından gerçekleştirilecek görevlerin işlevsel bir şekilde ayrıştırılmasını içerir.

Yani, en yüksek soyutlama düzeyinden başlayarak, tasarımcı ve müşteriler daha ileri düzeyde ayrıntı ortaya çıkarır.

Bu detaylı ayrıştırmayla, en düşük işlevsellik düzeyi (tek görev) elde edilir.

Örnek olarak, evcil hayvan mağazası POS sistemi için kısmi görev analizini göz önünde bulundurun.

Şekil 3.7'de gösterilmiştir.

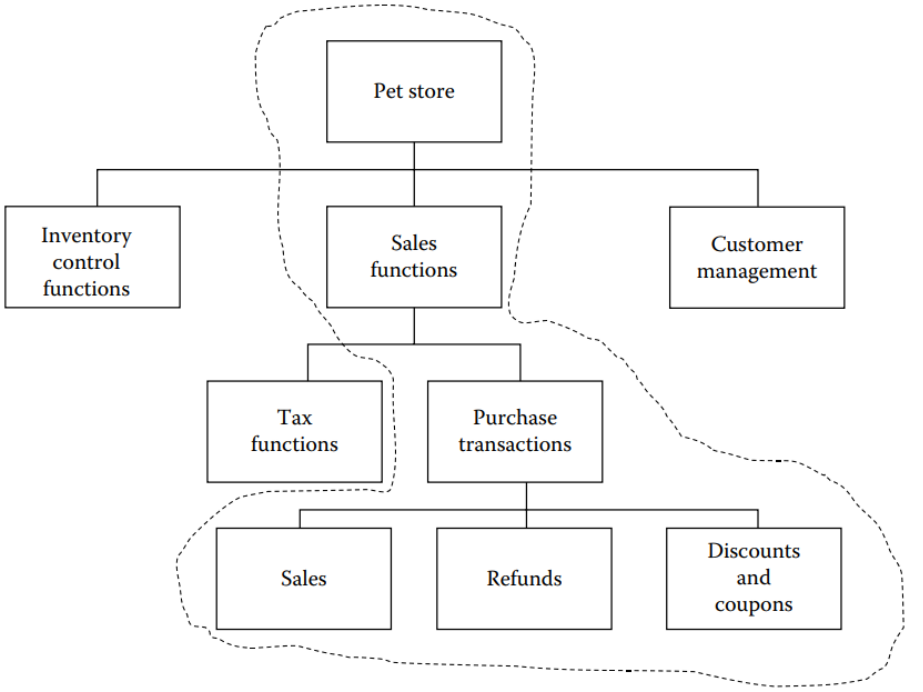


Figure 3.7 Partial task analysis for the pet store POS system.

Burada, kapsamlı evcil hayvan mağazası POS sisteminin üç ana bölümden oluştuğu kabul edilmektedir. Envanter kontrolü, satış ve müşteri yönetimi. Altında sondaj satış fonksiyonları, bunların aşağıdaki görevlerden oluştuğunu görüyoruz: vergi fonksiyonları ve satın alma işlemleri. Ardından satın alma işlemi fonksiyonuna geçilir, bu görevleri satış, geri ödeme, indirim ve kupon görevlerine ayırıyoruz.

Görev analizi ve ayrıştırma, yeterli bir ayrıntı düzeyine ulaşılana kadar devam eder (tipik olarak, bir yöntem veya ayrıştırılamaz prosedür düzeyine kadar) ve diyagram tamamlanır.

Kullanım Durumları

Kullanım senaryoları, daha karmaşık müşterilerin ve paydaşların isteklerini tanımlamanın bir yoludur. Kullanım durumları, sistem ve çevre arasındaki etkileşimleri, özellikle insan kullanıcıları ve diğer sistemler arasındaki etkileşimleri betimler.

Saf yazılım veya hibrit donanım yazılım sistemlerinin davranışını modellemek için kullanılabilirler.

Kullanım senaryoları, sistemin çalışma senaryolarını tasarımcının (müşterilerin aksine) bakış açısıyla açıklar. Kullanım durumları, tipik olarak, sistemin dış ortamıyla etkileşimlerini gösteren bir kullanım durumu diyagramı kullanılarak temsil edilir. Kullanım durumu diyagramında kutu, sistemin kendisini temsil eder. Çubuk figürler, sistemle etkileşime giren harici varlıkları belirleyen "aktörleri" temsil eder. Aktörler insanlar, diğer sistemler veya cihaz girdileri olabilir. İç elipsler, aktörlerin her biri için her bir kullanım etkinliğini temsil eder (kullanım durumları). Kesintisiz çizgiler, aktörleri her kullanımla ilişkilendirir. Şekil 3.8, bagaj muayene sistemi için bir kullanım durumu şemasını göstermektedir.

Üç kullanım gösterilmektedir: bagajın bir görüntüsünün alınması ("görüntü bagajı"), bir güvenlik tehdidinin tespiti (bu durumda torba, çevrimdışı işleme için konveyörden reddedilir) ve ardından sistem mühendisi tarafından yapılandırma. Görüntüleme kamerasının, ürün sensörünün ve reddetme mekanizmasının insan benzeri bir çubuk figürle temsil edildiğine dikkat edin - bu tipik bir durumdur - çubuk figür, insan olsun ya da olmasın bir sistem "aktörünü" temsil eder.

Her bir kullanım durumu, dikkate alınan sistemin çalışma senaryolarının yanı sıra ön ve son koşullar ve istisnaları açıklayan bir belgeleme biçimidir. Yinelemeli bir geliştirme yaşam döngüsünde, analiz ve tasarım iş akışları ilerledikçe bu kullanım durumları giderek daha iyi hale getirilecek ve ayrıntılı hale gelecektir.

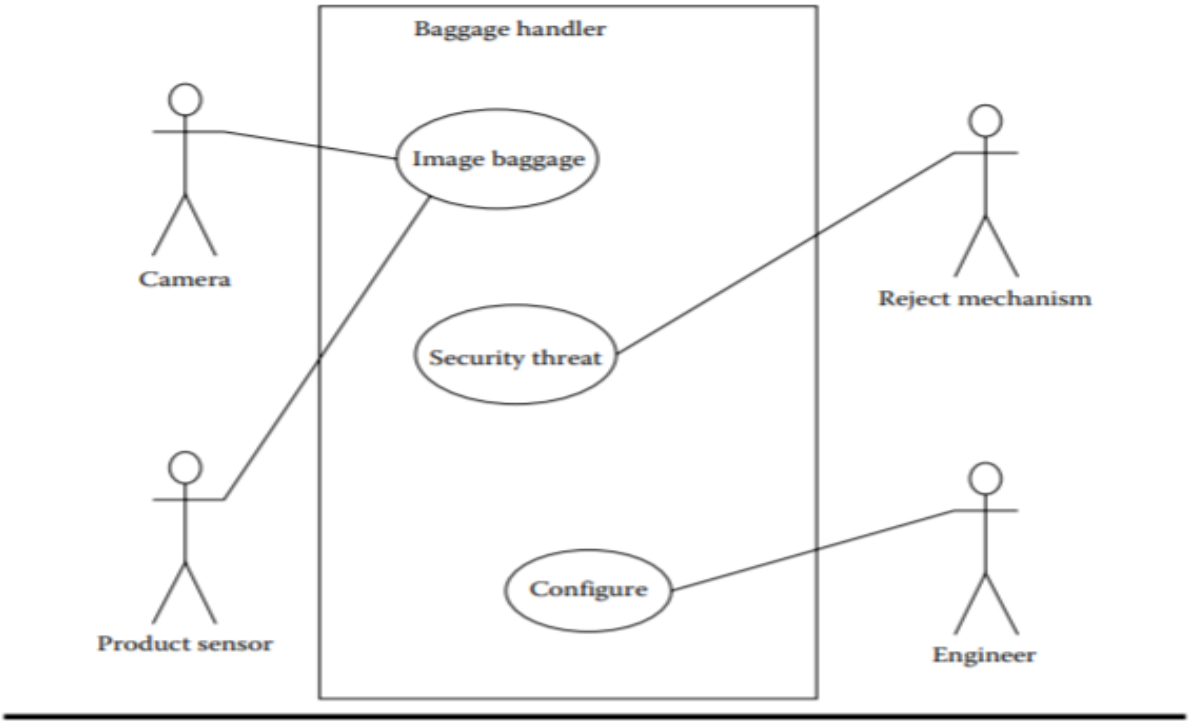


Figure 3.8 Use case diagram of baggage inspection system.

Daha sonra, her bir kullanım durumu tarafından tanımlanan davranışları tanımlamak için etkileşim diyagramları oluşturulur. İlk yinelemede, bu diyagramlar sistemi bir "kara kutu" olarak tasvir eder, ancak alan modellemesi tamamlandıktan sonra, kara kutu daha sonra görüleceği gibi nesnelerin bir işbirliğine dönüştürülür.

İyi geliştirilmişse, bazen kullanım durumları bir kalıp dili oluşturmak için kullanılabilir ve bu kalıplar ve türetilen tasarım öğeleri ilgili sistemlerde yeniden kullanılabilir (Issa ve Al-Ali 2010). Gereksinimleri belirtirken kalıpları kullanmak, daha yüksek düzeyde tutarlılık sağlar ve belirli gereksinim özelliklerinin otomatik ölçümündeki hataları azaltabilir.

Kullanıcı Hikâyeleri

Kullanıcı hikâyeleri, ilk gereksinimlerin keşfi ve proje planlaması için kullanılan kısa konuşma metinleridir. Kullanıcı hikâyeleri, çevik metodolojilerle birlikte yaygın olarak kullanılmaktadır.

Kullanıcı hikâyeleri, sistemin kendileri için ne yapması gerektiğine göre müşteriler tarafından ve kendi "sesleri" ile yazılır. Kullanıcı hikâyeleri genellikle üçe beş inçlik bir kart üzerine yazılan iki ila dört cümleden oluşur. Yaklaşık 80 kullanıcı hikâyesi genellikle bir sistem artışı veya gelişimi için uygundur, ancak uygun sayı, uygulama boyutuna ve kapsamına ve kullanılacak geliştirme metodolojisine (örneğin, çevik veya artımlı) bağlı olarak büyük ölçüde değişecektir.

Evcil hayvan mağazası POS sistemi için bir kullanıcı hikâyesi örneği şu şekildedir:

- Her müşteri bir kasada kolayca ödeme yapabilmelidir.
- Self servis desteklenecektir.
- Tüm kuponlar, indirimler ve geri ödemeler bu şekilde ele alınmalıdır.

Kullanıcı hikâyeleri, yalnızca hikâyenin uygulanmasının ne kadar süreceğine dair makul derecede düşük riskli bir tahmin yapmak için yeterli ayrıntı sağlamalıdır. Uygulama zamanı geldiğinde, hikâye geliştiricileri detayları detaylandırmak için müşteriyle buluşacak.

Kullanıcı hikâyeleri ayrıca kabul testinin temelini oluşturur. Örneğin, kullanıcı hikâyesinin doğru şekilde uygulanıp uygulanmadığını doğrulamak için bir veya daha fazla otomatik kabul testi oluşturulabilir.

Bakış Açıları

Bakış açıları, farklı seçmenlerden gelen bilgileri düzenlemenin bir yoludur. Örneğin, bagaj taşıma sisteminde, aşağıdaki paydaşların her biri için sistemin farklı bakış açıları vardır:

- Bagaj taşıma personeli
- Gezginler
- Bakım mühendisleri

- Havalimanı yöneticileri
- Düzenleyici kurumlar

Bu paydaşların her birinin ihtiyaçlarını ve bu bakış açılarının ortaya çıkardığı çelişkileri kabul ederek, çeşitli yaklaşımlar kullanılarak çatışmalar uzlaştırılabilir.

Gerçek bakış açıları, iş alanından, süreç modellerinden, işlevsel gereksinim özelliklerinden, organizasyon modellerinden vb. çeşitli bilgileri içerir.

Sommerville ve Sawyer (1997), aşağıdaki bileşenlerin her bir bakış açısında olması gerektiğini önerdi:

- Belirtimde kullanılan gösterimi tanımlayan bir temsil stili
- "Bakış açısının ele aldığı ilgi alanı" olarak tanımlanan bir alan
- Tanımlanmış tarzda ifade edilen bir sistem modeli olan bir spesifikasyon
- Spesifikasyonun nasıl oluşturulacağını ve kontrol edileceğini tanımlayan bir süreç modeline sahip bir çalışma planı
- Spesifikasyonu oluştururken, kontrol ederken ve değiştirirken gerçekleştirilen eylemlerin bir izi olan bir çalışma kaydı

Bakış açısı analizi genellikle önceliklendirme, anlaşma ve gereksinimlerin sıralanması için kullanılır.

Atölyeler

En genel düzeyde çalıştaylar, gereksinim sorunlarını çözmek için herhangi bir paydaş toplantısıdır. Çalıştayları resmi ve gayri resmi olmak üzere iki tür olarak ayırt edebiliriz.

Resmi çalıştaylar iyi planlanmış toplantılardır ve genellikle sözleşmeyle zorunlu kılınan "teslim edilebilir" etkinliklerdir. Örneğin, DOD-MIL-STD-2167, birden fazla gerekli ve isteğe bağlı atölyeyi (kritik incelemeler) içeriyordu. Resmi bir atölye tarzının iyi bir örneği JAD'de somutlaştırılmıştır.

Gayri resmi çalıştaylar genellikle yüksek düzeyde yapılandırılmış toplantılardan daha az sıkıcıdır. Ancak gayriresmi toplantılar çok özensiz olma eğilimindedir ve yanlış güvenlik duygusuna ve bilgilerin kaybolmasına neden

olabilir. Bir tür çalıştay gerekiyorsa, daha önce tartışılan başarılı toplantılar için parametreler kullanılarak resmi bir çalıştay yapılması tavsiye edilir.

İŞLEVSEL OLMAYAN GEREKSİNİMLERİ ORTAYA ÇIKARMA

İşlevsel olmayan gereksinim (NFR) ortaya çıkarma teknikleri, işlevsel gereksinimler ortaya çıkarma tekniklerinden farklıdır. NFR'ler genellikle gereksinim analizi sırasında gayri resmi olarak belirtilir, genellikle çelişkilidir ve yazılım geliştirme sürecinde uygulanması ve doğrulanması zordur. Borg ve ark. (2003), farklı kuruluşlarda NFR ile ilgili sorunların köklerini belirlemeye yönelik görüşmeler gerçekleştirmiştir. Sonuç, NFR ile ilgili sorunların geliştirme sürecinin dört aşamasında ortaya çıktığıydı: ortaya çıkarma, belgeleme, yönetim ve test; ortaya çıkarma aşamasında NFR ihmali tüm geliştirme süreci boyunca yayıldığından, ortaya çıkarma NFR ile ilgili olası sorunların ana kaynağı olarak işaretlenir. Örneğin, nedenler şunlardır:

(i) belirli kısıtlamalar

(ii) NFR'ler birbiriyle çelişme eğilimindedir

(iii) FR'leri ve NFR'leri ayırmak, aralarındaki bağımlılıkları izlemeyi zorlaştırırken, tüm gereksinimler birbirine karıştırıldığında işlevsel ve işlevsel olmayan hususları ayırmak zordur.

Mevcut çeşitli yöntemler arasından NFR'leri ortaya çıkarmak, detaylandırmak ve belgelemek için bir yöntem seçmek kolay değildir. Aşağıdakiler, NFR ortaya çıkarma yöntemlerinin arzu edilen özellikleridir (Herrmann ve diğerleri, 2007):

1. Daha az deneyimli personel tarafından yöntem kullanımını kolaylaştırmak ve sonuçların tekrarlanabilirliğini desteklemek için rehberli bir süreç

2. Kalite güvencesini kolaylaştırmak için ölçülebilir NFR'lerin türetilmesi

3. Öğrenmeyi desteklemek ve yeniden çalışmayı önlemek için türetilmiş NFR'lerin eksiksizliğini desteklemek için eserlerin yeniden kullanımı

4. Gizli gereksinimleri de yakalamak ve böylece eksiksizliği desteklemek için kalitenin sezgisel ve yaratıcı şekilde ortaya çıkarılması

5. Dengeleme kararlarını desteklemek için verimli ortaya çıkarma ve NFR önceliklendirme için odaklanmış çaba

6. Takas kararlarını desteklemek için NFR'ler arasındaki bağımlılıkların ele alınması

7. NFR'lerin fonksiyonel gereksinimlerle entegrasyonu

NFR'leri keşfetmenin yaygın yolları, sistem niteliklerinin rekabetçi analizini içerir: NFR'ler, pazardaki rakip ürünlerin nitelikleri analiz edilerek keşfedilebilir. Örneğin, rakip bir ürün için tepki süresi nedir? Ve daha iyisini mi yapmamız gerekiyor?

NFS'yi keşfetmek için başka bir teknik, bir gereksinim mühendisinin paydaşlara ve geliştirme ekibine sorulacak bir anket geliştirdiği önceden oluşturulmuş bir anket kullanmaktır.

Örneğin: “Sistem giriş hatalarına nasıl yanıt vermelidir? Sistemin hangi bölümleri daha sonra değiştirilmeye aday olabilir? Sistemin hangi verileri güvenli olmalı?” Bu sorular, standarttaki her bir NFR türü hakkında odaklanmak ve soru sormak için bazı şablon veya standartlar (örneğin, ISO 9126) izlenerek hazırlanabilir.

AÇIKLAMA ÖZETİ

Bu tur, birçok ortaya çıkarma tekniği içeriyor ve her birinin yol boyunca tartışılan avantajları ve dezavantajları var. Açıkça, bu tekniklerin bazıları çok genel, bazıları çok özel, bazıları paydaş bilgisine çok fazla güveniyor, bazıları yetersiz vb. Bu nedenle, gereksinimlerin ortaya çıkarılması sorununu başarılı bir şekilde ele almak için bazı teknik kombinasyonlarına ihtiyaç olduğu açıktır.

Gereksinim Ortaya Çıkarma Tekniklerinden Hangi Kombinasyonu Kullanılmalıdır?

Gereksinim belirleme tekniklerinin uygun bir karışımını seçme konusunda rehberlik sağlayacak çok az araştırma vardır. Kayda değer bir istisna, kullanıcıları bir kütüphaneden bir gösterim kombinasyonu seçmeye yönlendiren gereksinim mühendisliği tekniklerinin seçimine (KASRET) yönelik bilgiye dayalı yaklaşımdır (Eberlein ve Jiang 2011). Teknikler kütüphanesi ve

atama algoritması, bir literatür incelemesine ve endüstriyel ve akademik uzmanların anketine dayanmaktadır. Bununla birlikte, teknik henüz yaygın olarak kullanılmamıştır.

Uygun ortaya çıkarma teknikleri hakkında bir miktar rehberlik sağlamak için, önce daha önce tartışılan teknikleri, tekniklerin ortaya çıkarması muhtemel bilgi türlerine dayalı olarak kategoriler veya denklik sınıfları halinde kümeleriz. Sınıflar (görüşmeler, alan odaklı, grup çalışması, etnografi, prototip oluşturma, hedefler, senaryolar, bakış açıları) ve dahil edilen ortaya çıkarma teknikleri Tablo 3.2'de gösterilmiştir.

Şimdi, Tablo 3.3'te gösterildiği gibi (Zowghi ve Coulin'in çalışmasına dayalı olarak 1998) ortaya çıkarma sürecinin çeşitli yönleriyle başa çıkmada çeşitli tekniklerin ne kadar etkili olduğunu özetleyebiliriz.

Örneğin, görüşmeye dayalı teknikler, gereksinimlerin ortaya çıkarılmasının tüm yönleri için faydalıdır (ancak çok zaman alıcıdır). Öte yandan, prototipleme teknikleri, paydaşları analiz etmek ve gereksinimleri ortaya çıkarmak için en iyi şekilde kullanılır. Etnografik teknikler, problem alanını anlamak, paydaşları analiz etmek, gereksinimleri talep etmek vb. için iyidir.

Son olarak, bu ortaya çıkarma teknikleri (kümeler) arasında, bazılarının aynı şeyi başarması ve dolayısıyla birbirinin alternatifi olması bakımından açık bir örtüşme vardır. Diğer durumlarda, bu teknikler birbirini tamamlar. Tablo 3.4'te, alternatif (A) ve tamamlayıcı (C) ortaya çıkarma gruplamaları gösterilmektedir.

Uygun bir ortaya çıkarma teknikleri seti seçmenizde size rehberlik etmesi için Tablo 3.2 ila 3.4'ü kullanabilirsiniz. Kullanılacak bir dizi teknik seçerken, bir dizi tamamlayıcı eknik seçersiniz. Örneğin, bakış açısı analizi ve bir tür prototiplemenin bir kombinasyonu arzu edilir. Tersine, hem bakış açısı analizini hem de senaryo oluşturmayı kullanmak, muhtemelen aşırı derecede gereksiz bilgi verecektir.

Table 3.2 Organizing Various Elicitation Techniques Roughly by Type

Technique Type	*Techniques*
Domain-oriented	Card sorting Designer as apprentice Domain analysis Laddering Protocol analysis Task analysis
Ethnography	Ethnographic observation
Goals	Goal-based approaches QFD
Group work	Brainstorming Group work JAD Workshops
Interviews	Interviews Introspection Questionnaires
Prototyping	Prototyping
Scenarios	Scenarios Use cases User stories
Viewpoints	Viewpoints Repertory grids

Source: Zowghi, D., & Coulin, C., Requirements elicitation: A survey of techniques, approaches, and tools, in A. Aurum & C. Wohlin (Eds.), *Engineering and Managing Software Requirements*, pp. 19–46. Springer, 2005.

Örnek olarak, bir hastanedeki insanları izlemek için bir IoT sağlık sistemi örneğini düşünün. Burada, sistemin genel hedeflerini ve istenen sonuçları tanımlamak için bir girişimle başlamak uygun olacaktır. Ardından, sistem için geçerli olan yasaları, düzenlemeleri ve standartları keşfetmek için bir alan analizi kullanırdık. Kullanıcı hikayeleri, senaryolar ve röportajlar, sistem kullanıcılarından gereksinimlerin ortaya çıkarılması için uygun olacaktır. Gereksinim bulma ve iyileştirme süreci boyunca kuşkusuz bazı prototiplemeler olacaktır.

Table 3.3 Techniques and Approaches for Elicitation Activities

	Interviews	*Domain*	*Group work*	*Ethnography*	*Prototyping*	*Goals*	*Scenarios*	*Viewpoints*
Understanding the domain	•	•	•	•		•	•	•
Identifying sources of requirements	•	•	•			•	•	•
Analyzing the stakeholders	•	•	•	•	•	•	•	•
Selecting techniques and approaches	•	•	•					
Eliciting the requirements	•	•	•	•	•	•	•	•

Source: Zowghi, D., & Coulin, C., Requirements elicitation: A survey of techniques, approaches, and tools, in A. Aurum & C. Wohlin (Eds.), *Engineering and Managing Software Requirements*, pp. 19–46, Springer, 2005.

Table 3.4 Complementary and Alternative Techniques

	Interviews	*Domain*	*Groupwork*	*Ethnography*	*Prototyping*	*Goals*	*Scenarios*	*Viewpoints*
Interviews		C	A	A	A	C	C	C
Domain	C		C	A	A	A	A	A
Groupwork	A	C		A	C	C	C	C
Ethnography	A	A	A		C	C	A	A
Prototyping	A	A	C	C		C	C	C
Goals	C	A	C	C	C		C	C
Scenarios	C	A	C	A	C	C		A
Viewpoints	C	A	C	A	C	C	A	

Source: Zowghi, D., & Coulin, C., Requirements elicitation: A survey of techniques, approaches, and tools, in A. Aurum & C. Wohlin (Eds.), *Engineering and Managing Software Requirements*, pp. 19–46, Springer, 2005.

Gereksinim keşfi ilerledikçe bu tekniklerin her biri resmi olmayandan daha titiz hale gelecektir. Bu yaklaşım genellikle federal, eyalet ve belediye yönetimleri ve hatta büyük endüstriyel projeler için kullanılır.

Bununla birlikte, ortaya çıkarma tekniklerinin "gümüş kurşun" kombinasyonu yoktur. Doğru karışım, uygulama alanına, müşteri organizasyonunun kültürüne ve gereksinim mühendisinin kültürüne, projenin boyutuna ve diğer birçok faktöre bağlı olacaktır. Bu konuda deneyimlerden ders almak çok önemlidir.

Gereksinim Belirleme Tekniklerinin Yaygınlığı

Bu tartışmayı bitirmeden önce, endüstride çeşitli ortaya çıkarma tekniklerinin nasıl yaygın olarak kullanıldığına dair bir fikir edinelim. Bunu yapmak için yazılım profesyonellerinin anketine dönüyoruz (Kassab ve diğerleri 2014). Hangi gereksinim belirleme teknik(ler)ini kullanıyorsunuz sorusuna verilen cevapların bir özeti. Şekil 3.9'da gösterilmiştir. Yanıtlar, beyin fırtınası, görüşmeler ve prototip oluşturmanın en sık kullanılan bilgi toplama teknikleri olduğunu ortaya çıkardı (her biri yaklaşık %10). Bu bölümde tartışılan diğer teknikler nispeten seyrek kullanılmıştır.

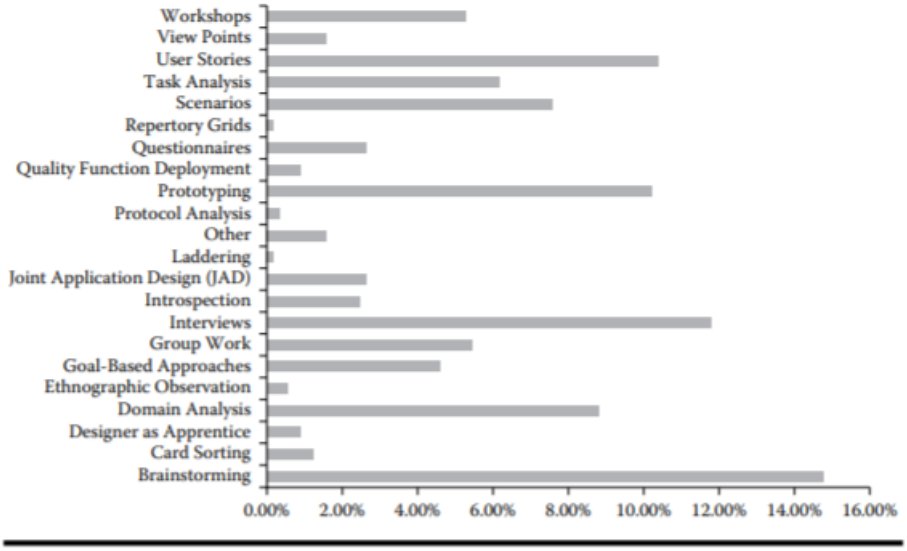


Figure 3.9 Summary of answers to the question "which requirements elicitation technique(s) do you use?" (Adapted from Kassab, M., Neill, C & Laplante, P. State of practice in requirements engineering: contemporary data. *Innovations in Systems and Software Engineering*, 10, no. 4 (2014): 235–241. With permission.)

TEHLİKELERİ ORTAYA ÇIKARMAK

Daha önce "yapılmaması gereken" davranışların, ortaya çıkan çıktı davranışları kümesi olduğunu belirtmiştik. İstenmeyen ve tehlikelerin neden olma eğiliminde olan davranışların bir alt kümesi olduğu ciddi veya yıkıcı arızalardı. "Ciddi" ve "felaket" terimleri özneldir ancak genellikle can kaybını, ciddi yaralanmayı, büyük altyapı hasarını, ya da büyük mali kaybı içerir. Örneğin, evcil hayvan mağazası POS' u için bazı "yapılmaması gereken" gereksinimler şunları içerir:

- Sistem, müşteri bilgilerini harici sistemlere ifşa etmeyecektir.
- Sistem yetkisiz erişime izin vermeyecektir.
- Sistem, müşterilerin mağaza kredisini aşmasına izin vermeyecektir.

Bu örneklerde, ilk iki gereksinim tehlike olarak kabul edilebilir, çünkü şirkete mali zarar verme potansiyeli üçüncüsünden çok daha fazladır.

Tehlikeler, doğal olarak meydana gelen girdi anormalliklerinin bir fonksiyonudur.(donanım arızaları gibi) veya yapay olarak meydana gelen (davetsiz misafirlerin saldırıları gibi) (Voas ve Laplante 2010). Bu anormal giriş olaylarının tanımlanması gerekir ve sistem için uygun bir "yapılmaması gereken" gereksinimleri seti geliştirmek için, bunların ortaya çıkan arıza modları ve kritikliği,

gereksinimlerin ortaya çıkarılması aşamasında belirlenmelidir. Diğer gerekliliklerde olduğu gibi, “yapılmaması gereken” gerekliliklerine öncelik verilmesi gerekir. Tehlike belirleme için tipik teknikler, kötüye kullanım vakalarının geleneksel gelişimini, antimodeling ve resmi yöntemleri içerir (Robinson 2010). Sistemin veya ilgili sistemlerin önceki sürümlerinden oluşturulan istenmeyen davranışların kontrol listeleri de istenmeyen davranışların belirlenmesinde yardımcı olur. Geçerli standartlar ve düzenlemeler ayrıca belirli "yapılmaması" gerekliliklerini de içerebilir, örneğin Amerika Birleşik Devletleri'nde Sağlık Sigortası Taşınabilirlik ve Sorumluluk Yasası (HIPPA 1996), tüm yargı alanlarında belirli kişisel bilgilerin yetkisiz taraflara ve standart inşaat kurallarına verilmesini yasaklar. Belirli inşaat uygulamalarına karşı çok sayıda yasak içerir.

Kötüye Kullanım Durumlar

Kullanım senaryoları, istenen davranışın yapılandırılmış, kısa açıklamalarıdır. İstenen davranışı tanımlayan kullanım durumları olduğu gibi, istenmeyen davranışları tanımlayan kötüye kullanım durumları (veya kötüye kullanım durumları) vardır.

Çoğu sistem için tipik suistimaller, güvenlik ihlallerini ve diğer kötü niyetli davranışların yanı sıra eğitimsiz, yönünü şaşırmış veya beceriksiz kullanıcılar tarafından kötüye kullanım. Cleland-Huang ve ark. (2016), tehdit modelleme ve beyin fırtınasına dayalı olarak kötüye kullanım vakalarını belirlemenin çeşitli yollarını açıklar.

Kullanım senaryoları oluşturmanın kolay bir yolu, istenmeyen bir sistem kullanıcısı olan istenmeyen bir kişi rolünü üstlenmek ve ardından böyle bir kişinin davranışlarını modellemektir (Cleland-Huang 2014). İstenmeyen kişiler, bilgisayar korsanlarını, davetsiz misafirleri, casusları ve hatta iyi niyetli, ancak beceriksiz kullanıcıları içerebilir.

Antimodeller

İstenmeyen davranışları elde etmenin başka bir yolu da sistem için antimodeller yaratmaktır. Antimodeller hata ağaçlarıyla ilgilidir, yani model, sistem hatasına yol açan istenmeyen davranışlar için bir sebep-sonuç hiyerarşisi yaratılarak türetilir. Ardından, sistem arızasının nedenleri “yapmayacak” gerekliliklerini oluşturmak için kullanılır. Örneğin, Şekil 3.10'da gösterilen hasarlı bagajın istenmeyen sonucunu içeren bagaj taşıma sistemi için güvenlik işlevselliğini düşünün.

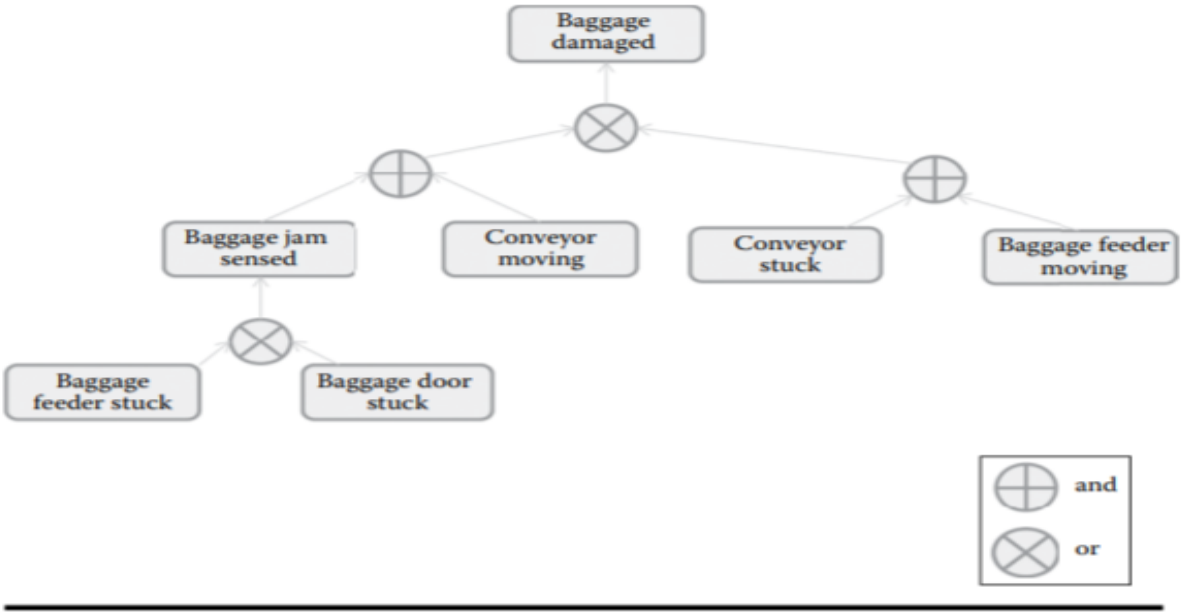


Figure 3.10 Partial antimodel for baggage handling system.

erir.

Şekil bizi aşağıdaki ham gereksinimleri yazmaya yönlendiriyor:

- Bagaj sıkışması algılanırsa konveyör hareket etmemelidir.
- Bagaj besleyici sıkışmışsa, konveyör hareket etmemelidir.
- Bagaj kapısı sıkışmışsa konveyör hareket etmemelidir.
- Konveyör sıkışmışsa, bagaj besleyici hareket etmemelidir.

Bu gereksinimlerin daha fazla analize ve muhtemelen basitleştirilmesine ihtiyacı var, ancak antimodel, bu ham gereksinimleri sistematik bir şekilde türetmemize yardımcı oldu.

Resmi Yöntemler

Bölüm 6'da tartışılacağı üzere, amaçları, işlemleri, gereksinimleri ve kısıtlamaları ile ilgili olarak sistem ve çevresinin bir modelini oluşturmak için matematiksel formalizmler kullanılabilir. Bu formalizmler daha sonra sistemin çeşitli özelliklerini incelemek ve istenmeyenlerin bulunmadığından emin olmak için otomatik model denetleyicileri ile birlikte kullanılabilir. Örneğin, davranışı modellemek için UML/SysML kullanılıyorsa, resmi etkinlik diyagramlarına güvenlik endişeleri eklenebilir. Aktivite diyagramları Ek C, UML'de açıklanmıştır.

VIGNETTE 3.1 Requirements Engineering for Safety-Critical Systems

Safety-critical systems are those in which failure can result in loss of human life or significant injury. These kinds of systems include public infrastructure for power and water, medical systems, transportation systems, and much more. With the growth of smart and autonomous systems, for example, smart highways, driverless vehicles, and robotic surgery, the focus on safety requirements has become paramount.

There are some domain-specific standards (e.g., nuclear, medical devices, aviation) and general standards that provide guidance for developing requirements for safety-critical systems. One example of a general standard is NASA-STD-8719.13, which provides guidance for safety considerations in software intensive systems (NASA 2013).

This standard uses a risk assessment matrix table to show the probability and severity of a particular hazard (Figure 3.11).

Standart, gereksinimlerin etkilerinin ve değişikliklerin değerlendirilmesini sağlamak için ihtiyaç duyulan mühendislik yaşam döngüsü boyunca izlenebilirlik için güvenlik gereksinimlerinin etiketlenmesini şart koşar. Benzersiz olarak tanımlanan tehlikeler, gereksinimler belgesindeki özel bir bölümde listelenebilir veya gereksinimin yanında bir bayrakla belirtilebilir veya bir gereksinim yönetim aracı içinde etiketlenebilir.

Systems severity levels	System hazard likelihood of occurrence				
	Improbable	Unlikely	Possible	Probable	Likely
Catastrophic	4	3	2	1	1
Critical	5	4	3	2	1
Moderate	6	5	4	3	2
Negligible	7	6	5	4	3

1 = Highest priority (highest system risk), 7 = Lowest priority (lowes system risk).

Figure 3.11 The system risk index matrix of NASA-STD-8719.13C. (From NASA [National Aeronautics and Space Administration], NASA-STD 8719.13, Software Safety Standard, 2013, http://NASA.gov [accessed January 2017].)

Yazılım güvenliği gereksinimleri, güvenli olmayan sistem davranışına karşı koruma sağlamaktan fazlasını yapabilir. Yazılım, sistemi izlemek, kritik verileri analiz etmek, eğilimleri aramak ve tehlikeli bir durumun habercisi olabilecek olaylar meydana geldiğinde sinyal vermek veya harekete geçmek için proaktif olarak kullanılabilir. Böyle bir gösterge tespit edildiğinde, yazılım bu tehlikenin etkilerini önlemek veya azaltmak için kullanılabilir. Kaçınma veya azaltma, sistemin tamamen veya kısmen geri yüklenmesini veya sistemin güvenli bir duruma getirilmesini içerebilir.

Hazırlayanlar Gece/B şubesinden;

205541002-Ahmet Kadir Aydın
205541026-Rana Mahsereci
205542016-Mahmut Başcı
195542004-Lütfiye Gül Yılmaz
195541060-Çağdaş Kalak